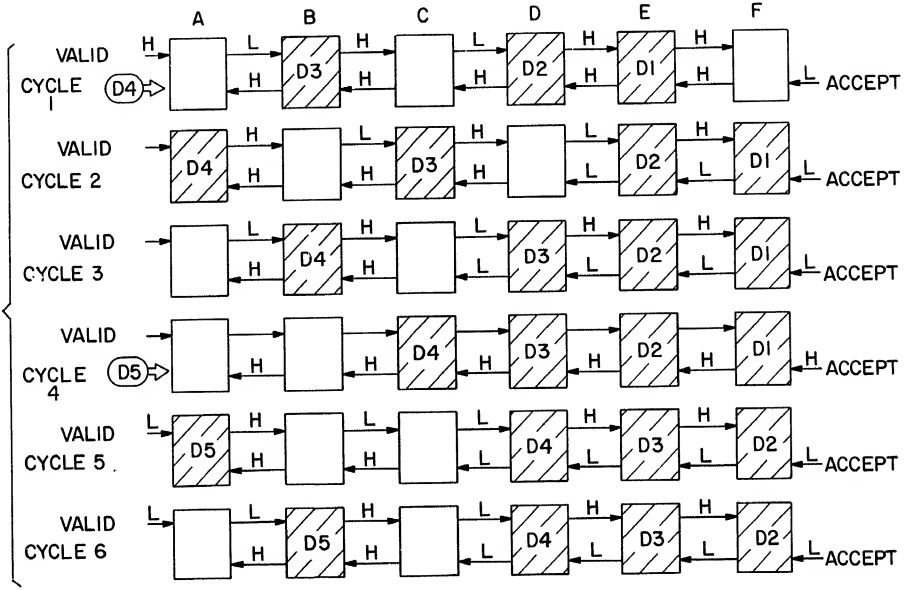


FIG. 1

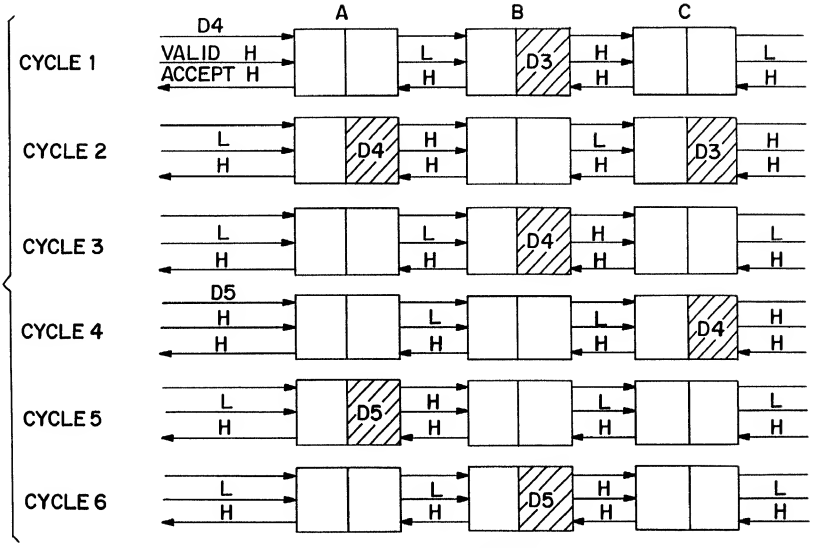


FIG. 2(A)

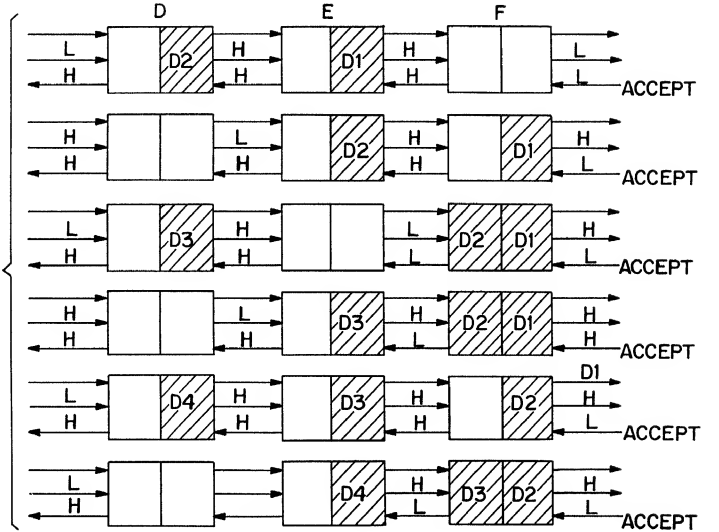
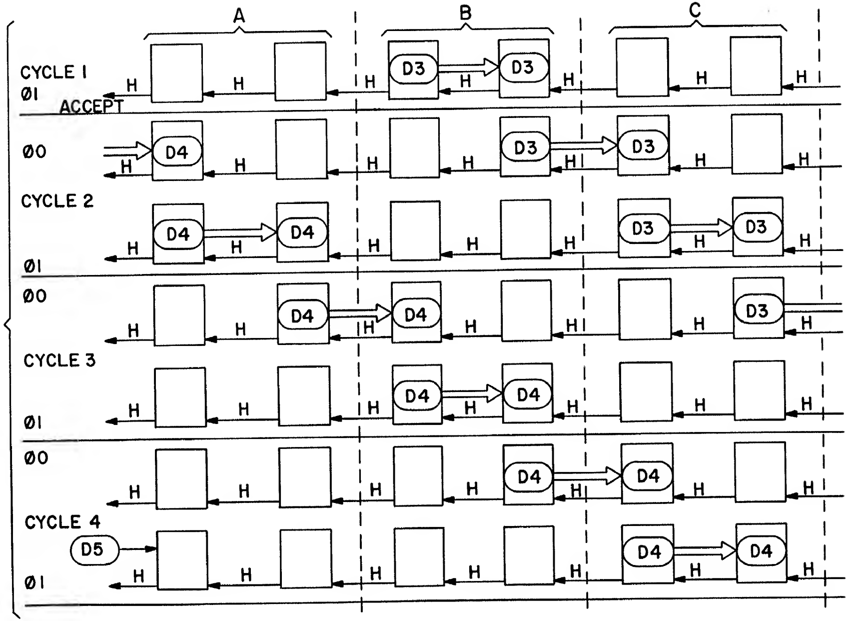


FIG. 2(B)



A
B
C
CYCLE 1
Ø1
ACCEPT
Ø0
CYCLE 2
Ø1
Ø0
CYCLE 3
Ø1
Ø0
CYCLE 4
Ø1
D3
D4
D5
H

FIG. 3A-1

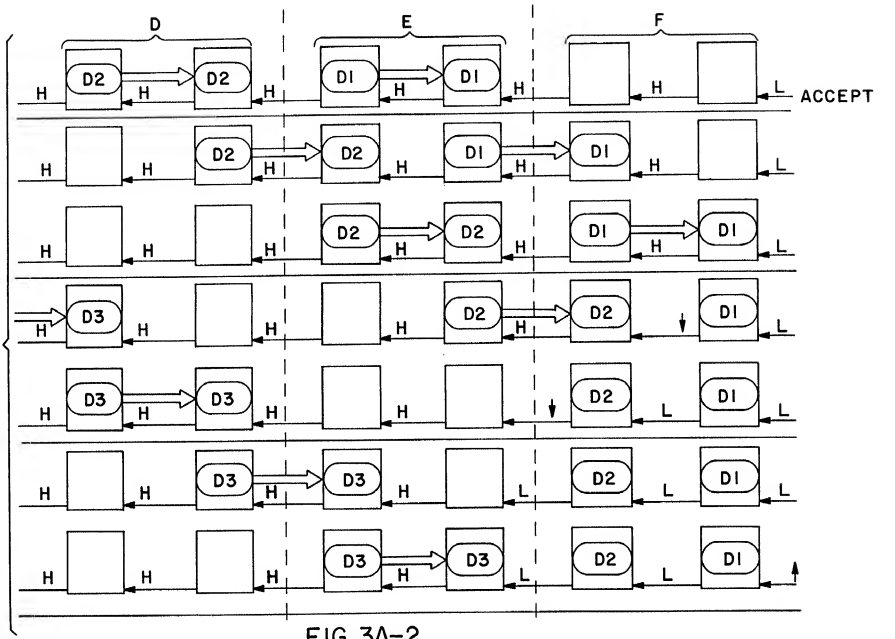


FIG. 3A-2

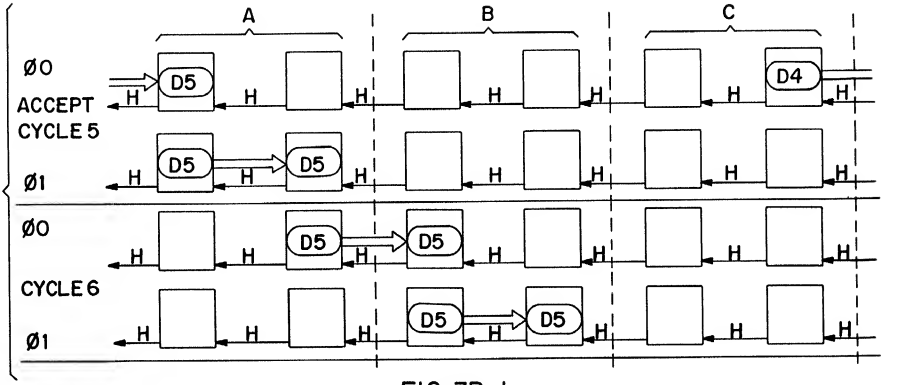


FIG. 3B-1

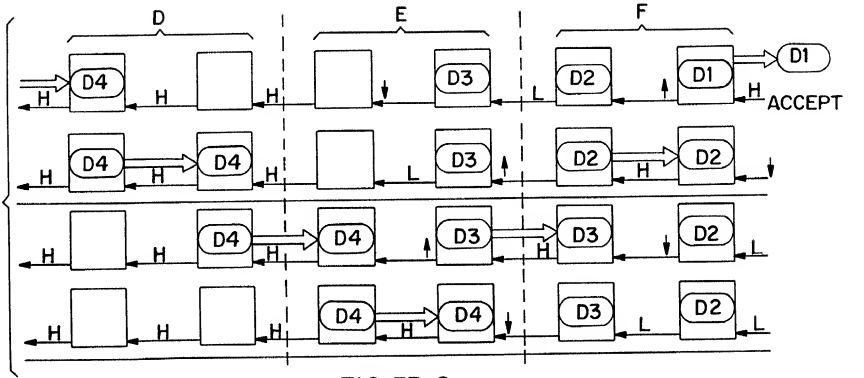


FIG. 3B-2

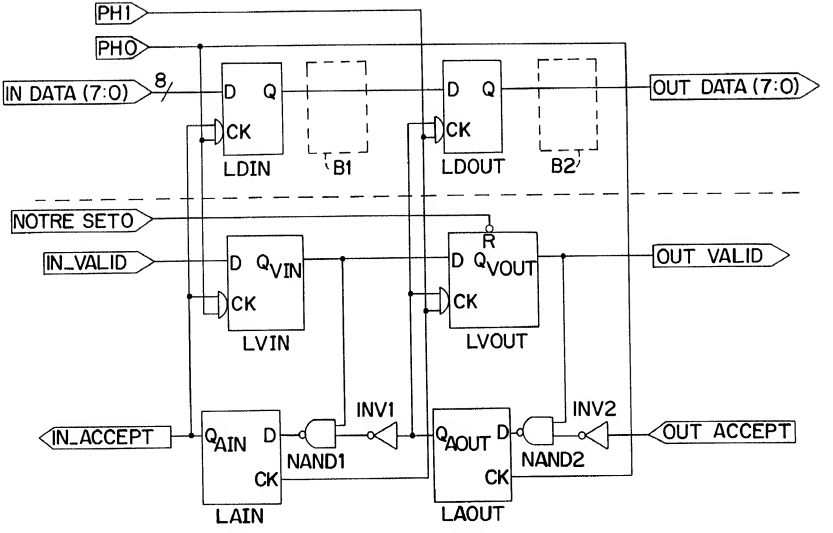


FIG. 4

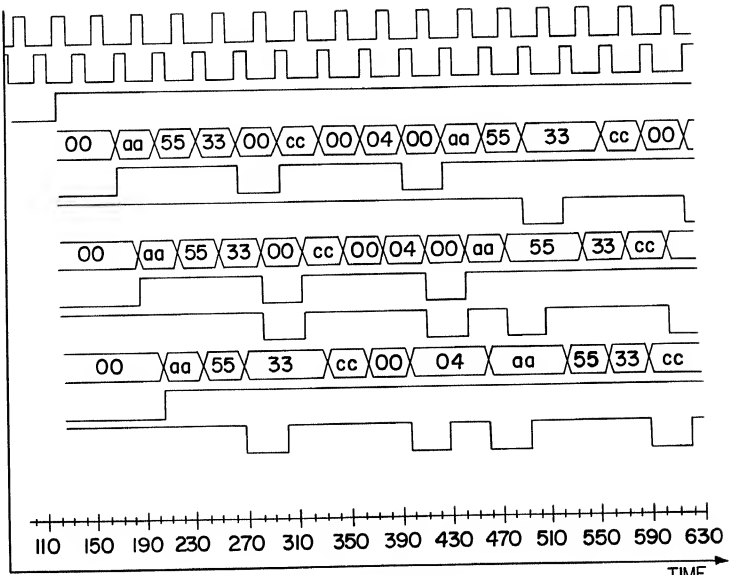


FIG. 5(A)

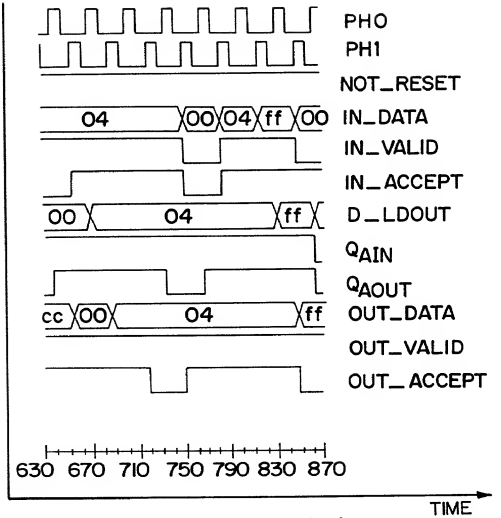


FIG. 5(B)

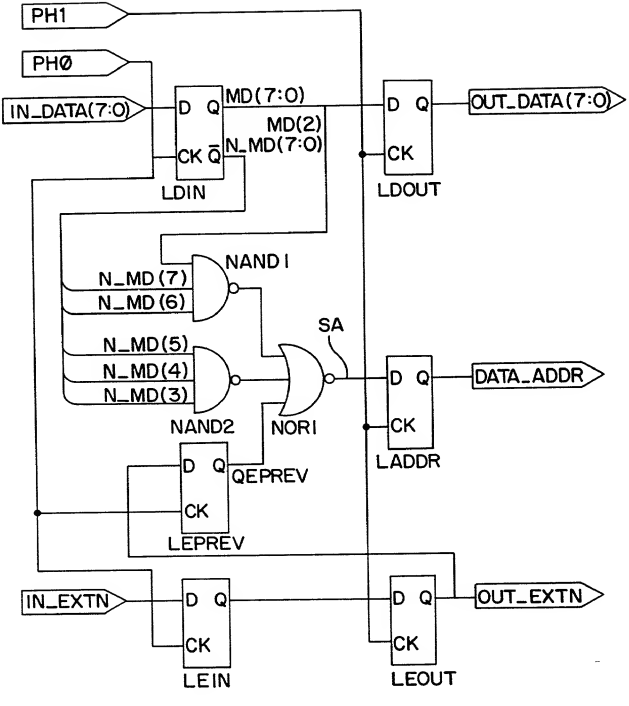


FIG. 6

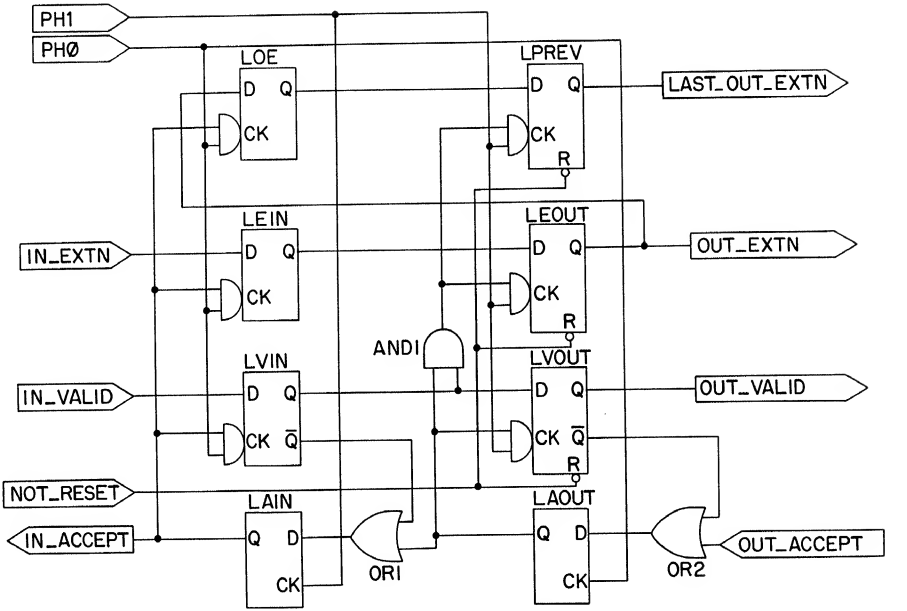


FIG. 7

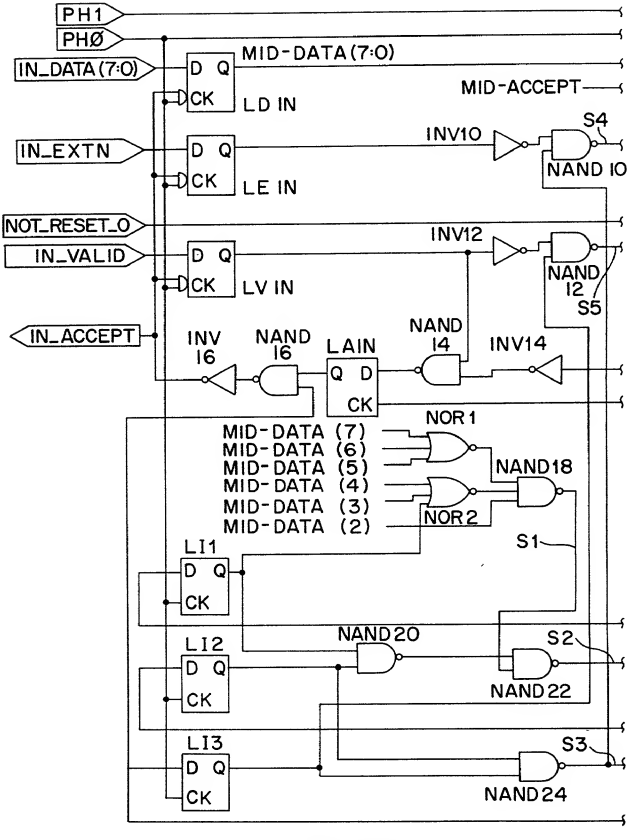


FIG. 8(A)

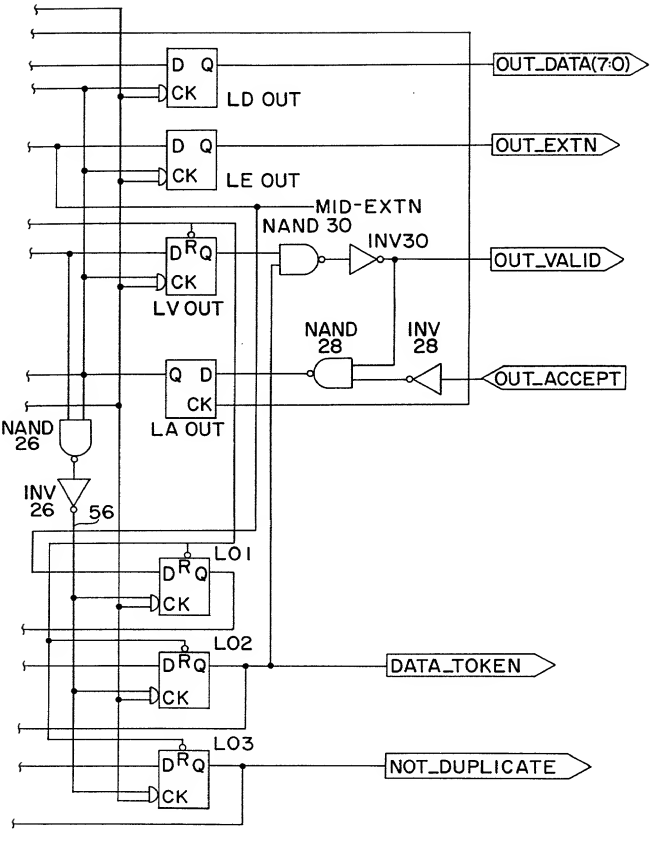


FIG. 8(B)

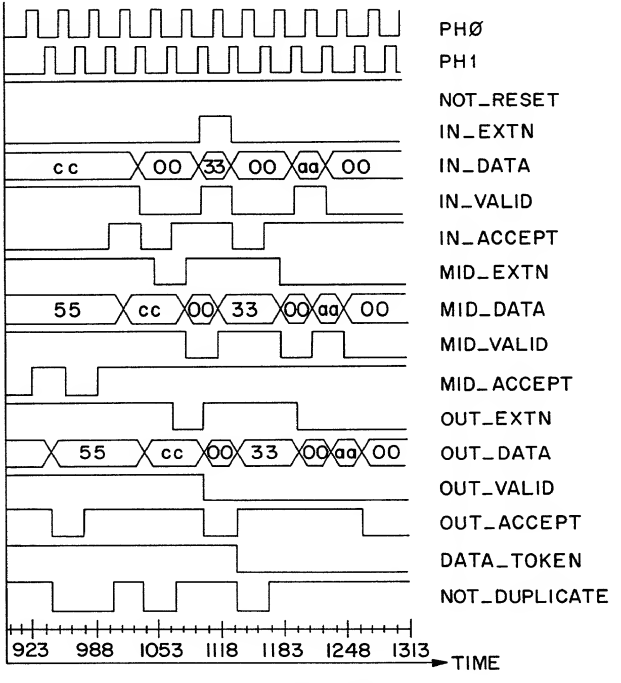


FIG. 9(B)

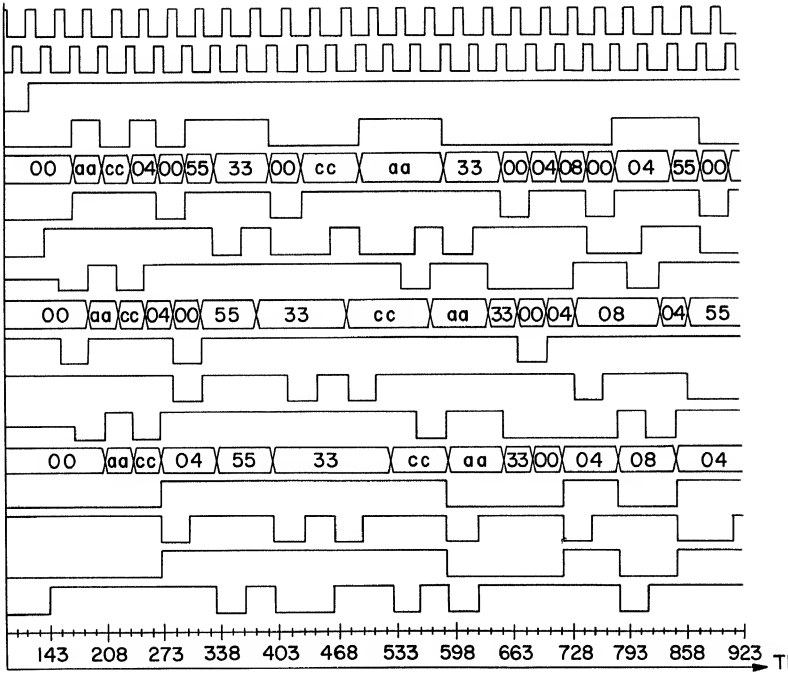


FIG. 9(A)

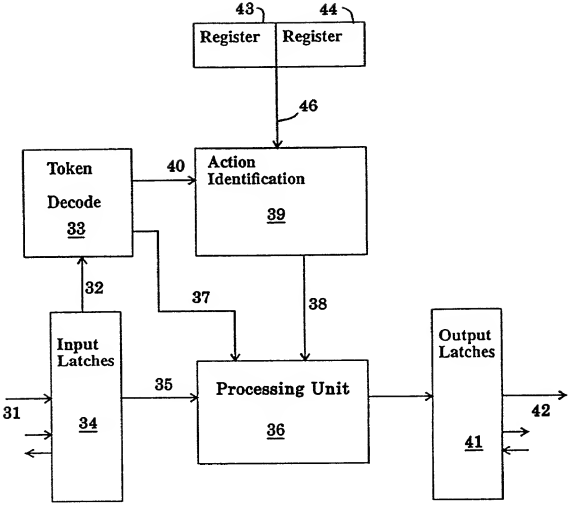


FIG. 10

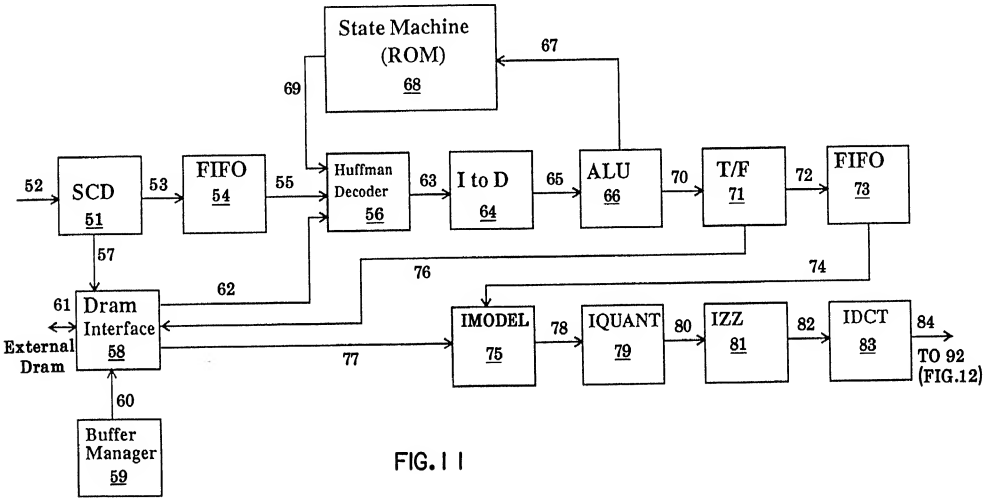


FIG. 11

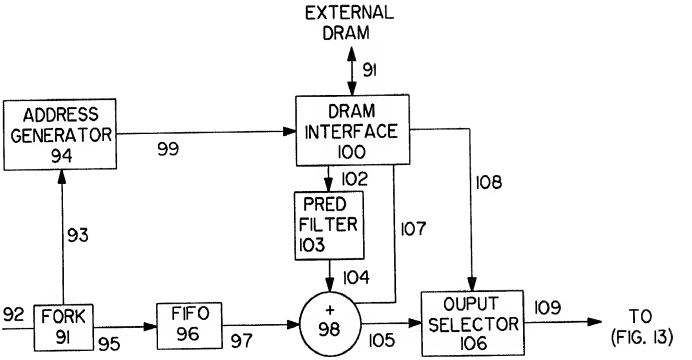


FIG. 12

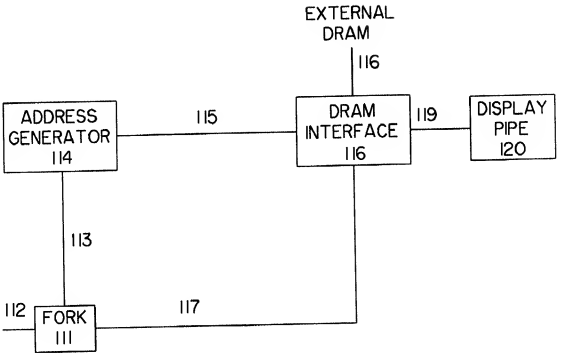


FIG. 13

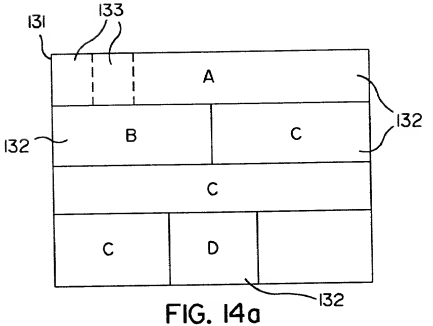


FIG. 14a

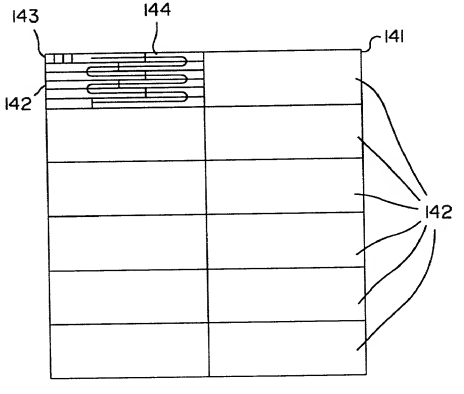


FIG. 14b

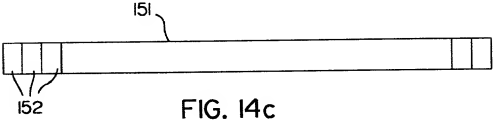


FIG. 14c

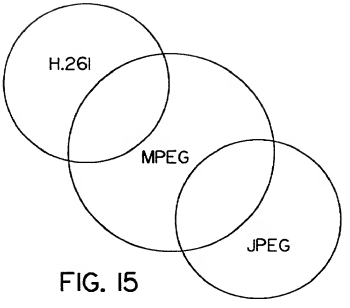


FIG. 15

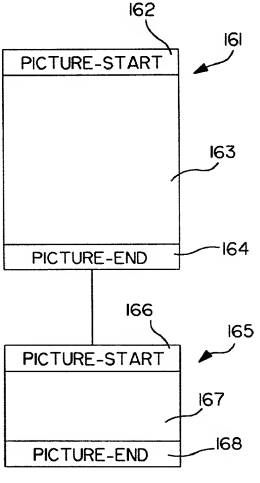


FIG. 16

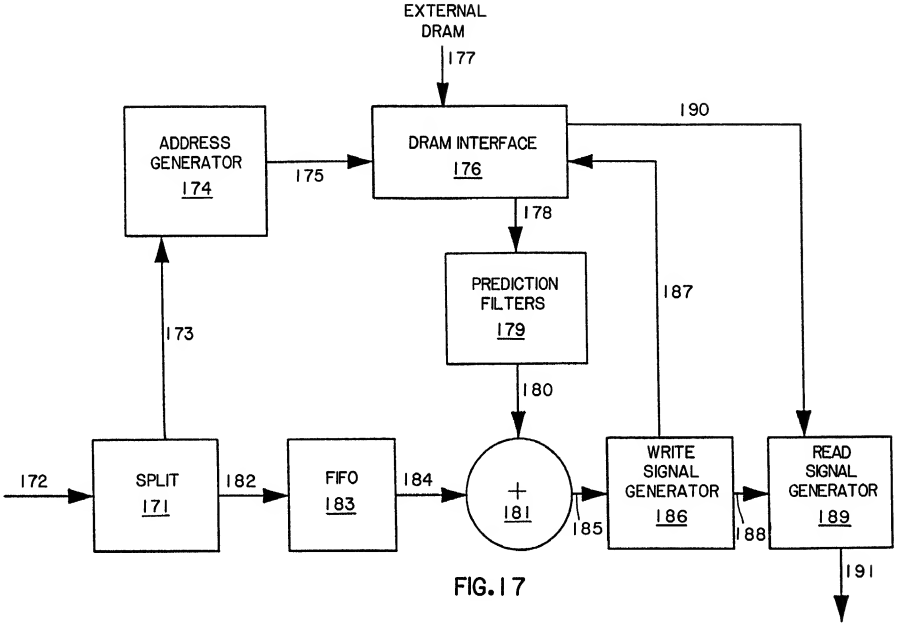


FIG. 17

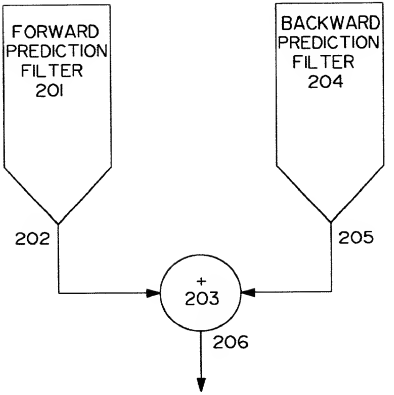


FIG. 18

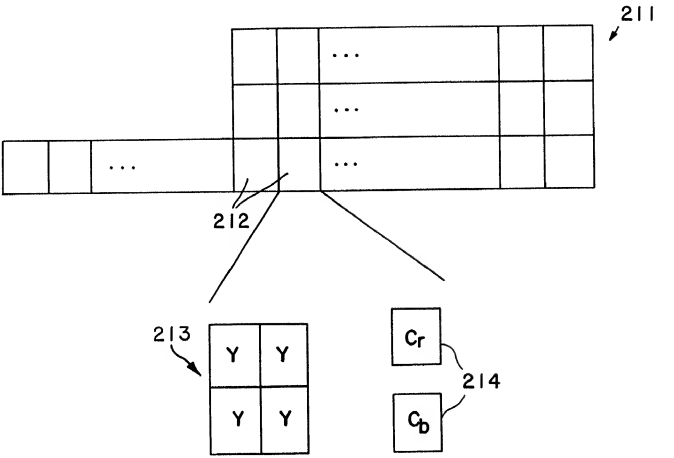


FIG. 19

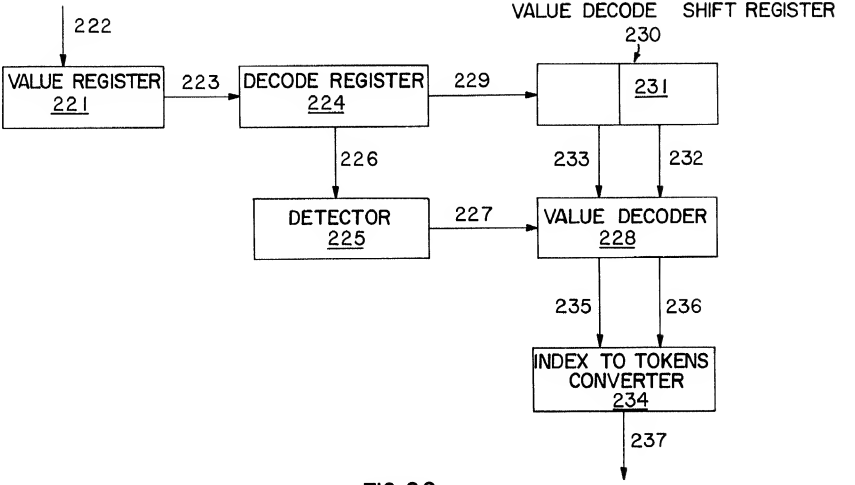


FIG.20

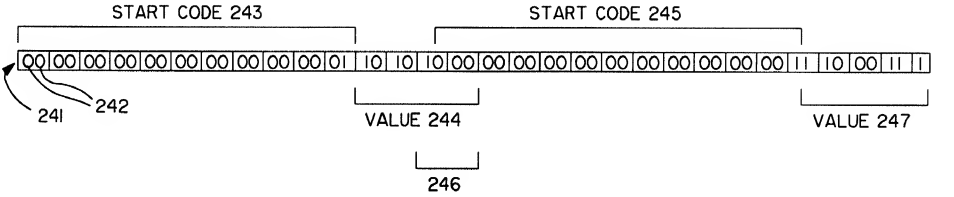


FIG. 21

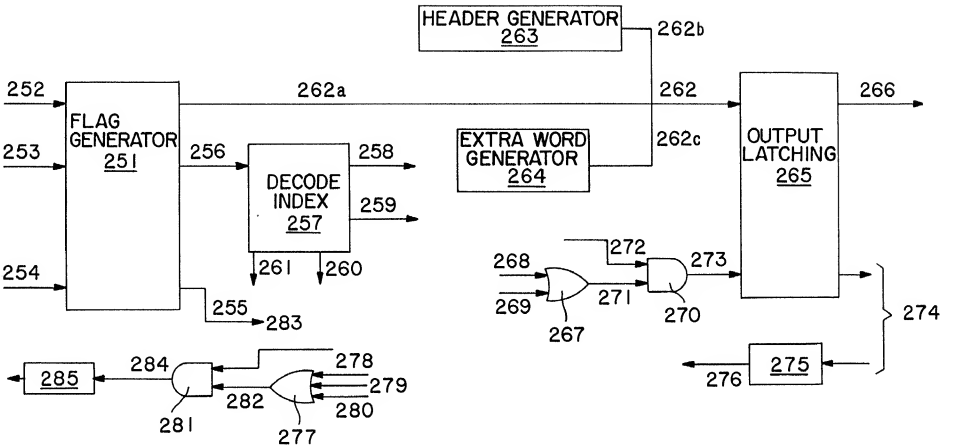


FIG.22

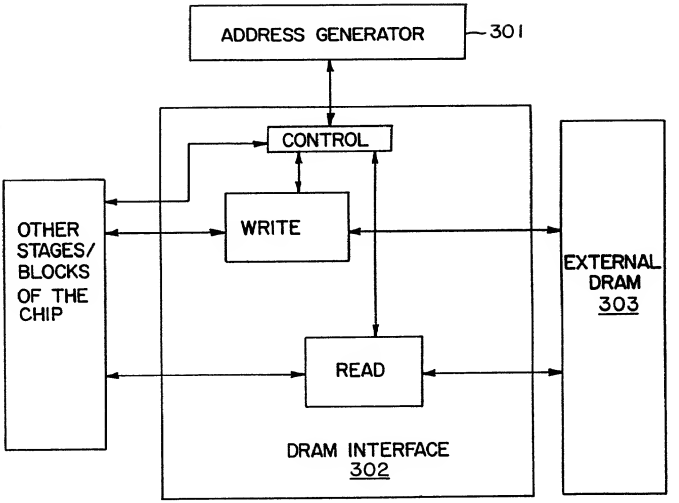


FIG.23

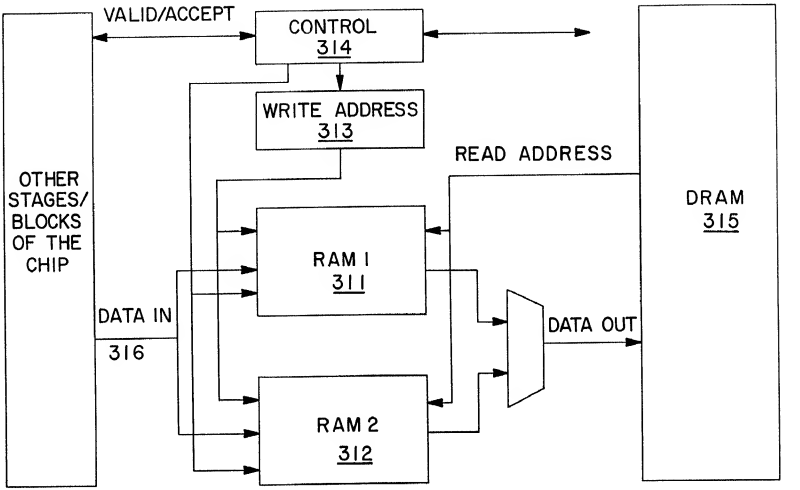


FIG.24

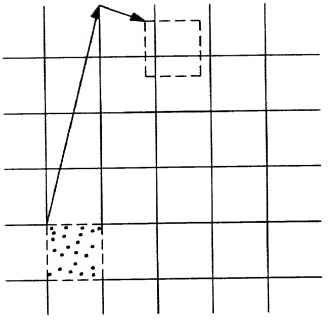


FIG. 25

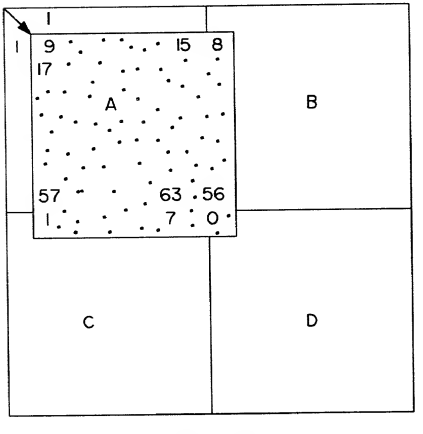


FIG. 26

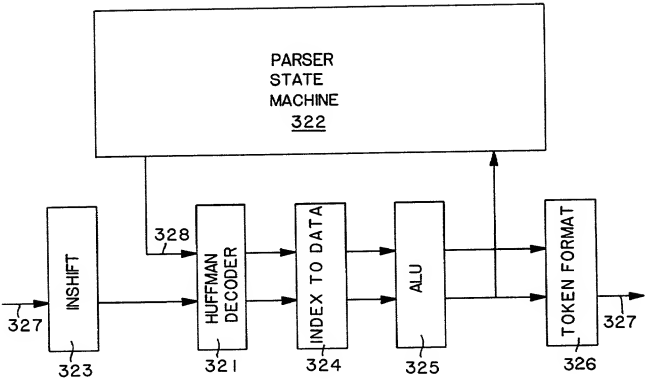


FIG.27

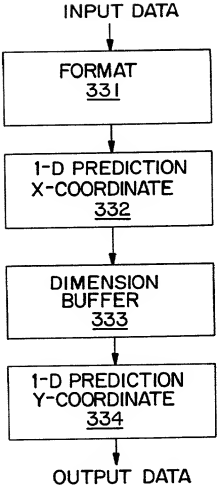


FIG.28

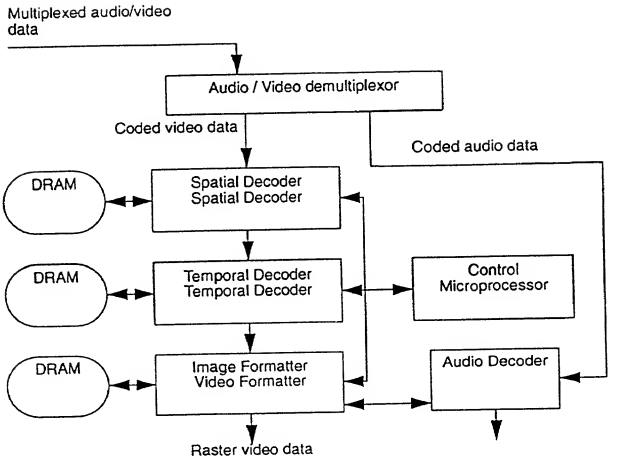


FIG.29

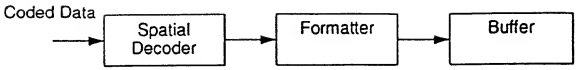


FIG.30

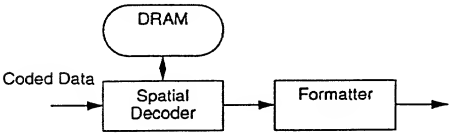


FIG.31

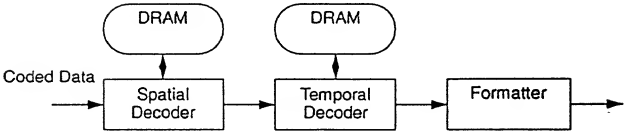


FIG.32

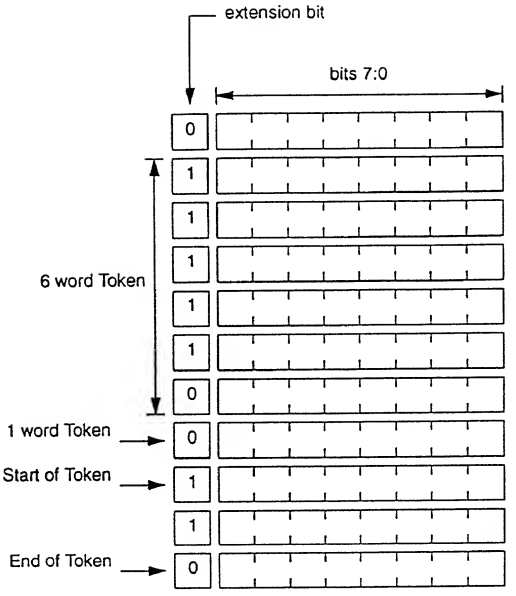


FIG.33

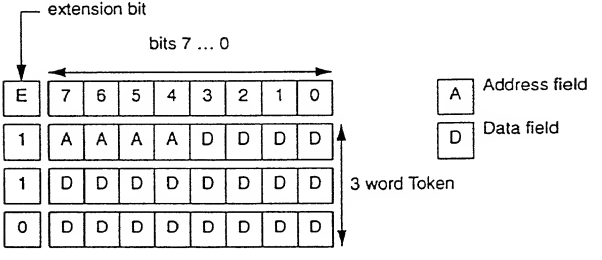


FIG.34

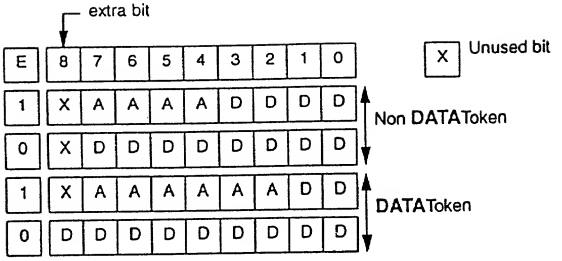
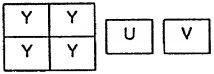
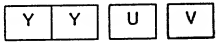


FIG.35



MPEG 4:2:0
macroblock

FIG.36A



JPEG 2:1:1
macroblock

FIG.36B

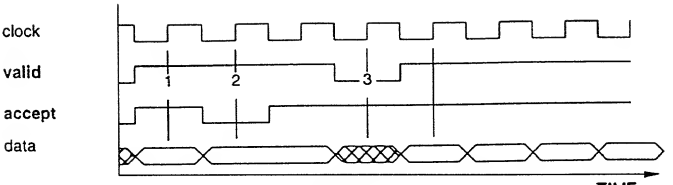

clock
valid
accept
data
1
2
3
TIME

FIG.37

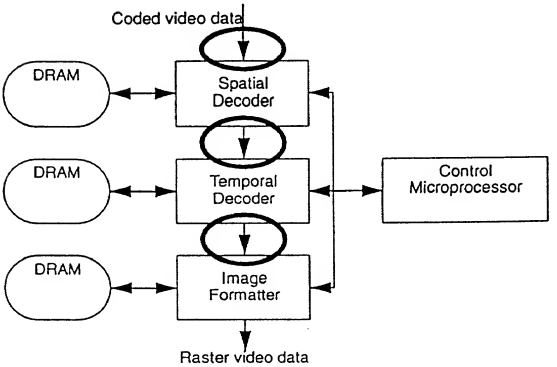

Coded video data
DRAM
Spatial Decoder
DRAM
Temporal Decoder
Control Microprocessor
DRAM
Image Formatter
Raster video data

FIG.38

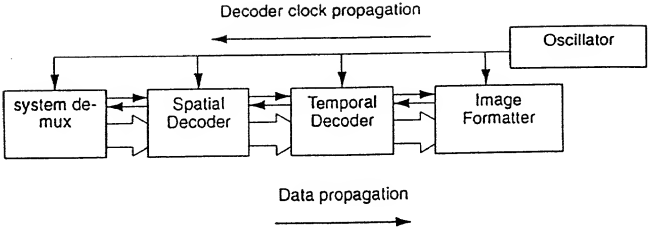


FIG.39

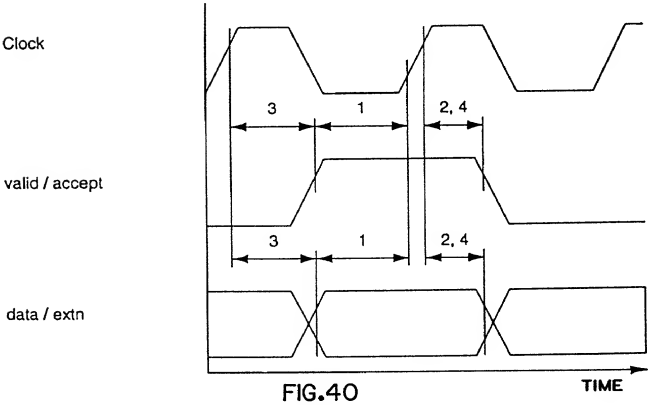


FIG.40


Access Start
Data Transfer
Default State

FIG.41

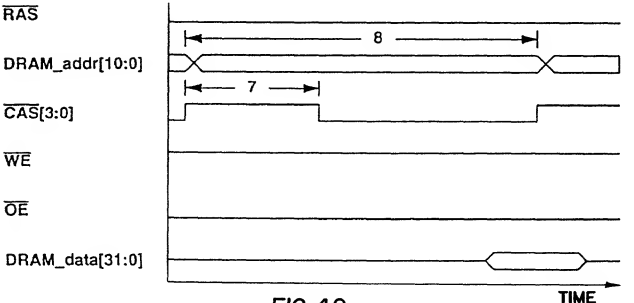

RAS
DRAM_addr[10:0]
CAS[3:0]
WE
OE
DRAM_data[31:0]
8
7
TIME

FIG.42

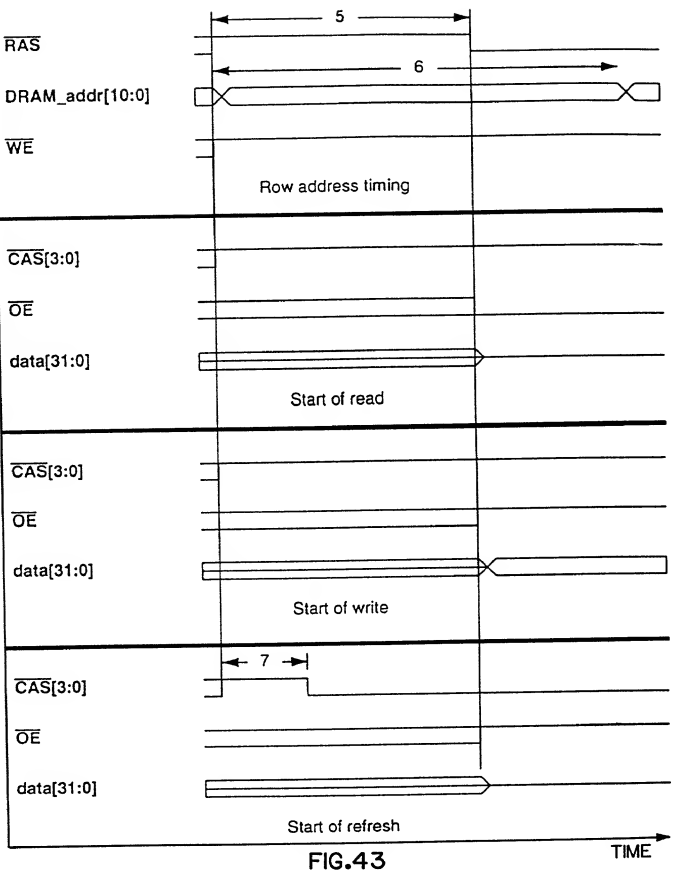


FIG.43

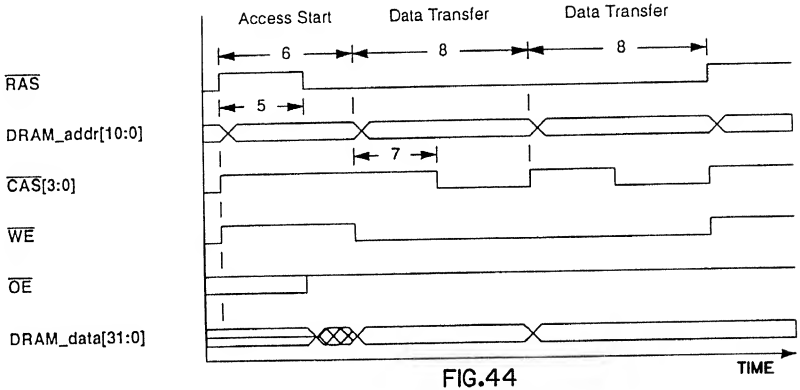


FIG.44

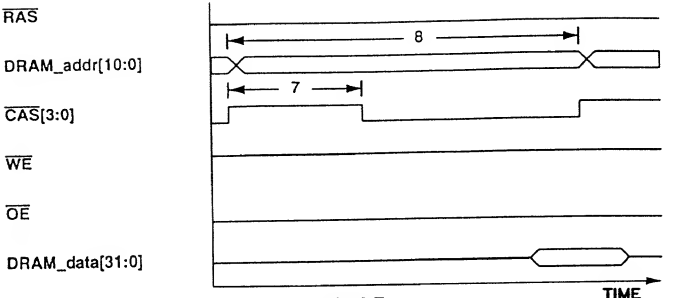


FIG.45

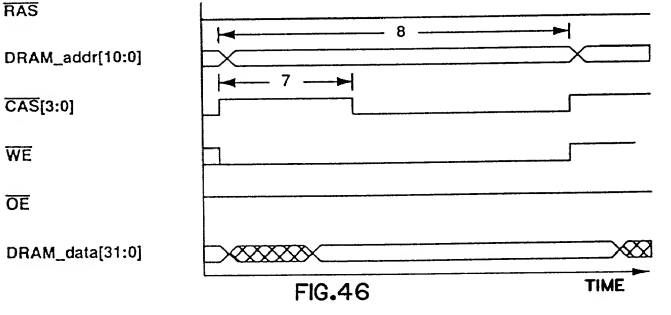


FIG.46

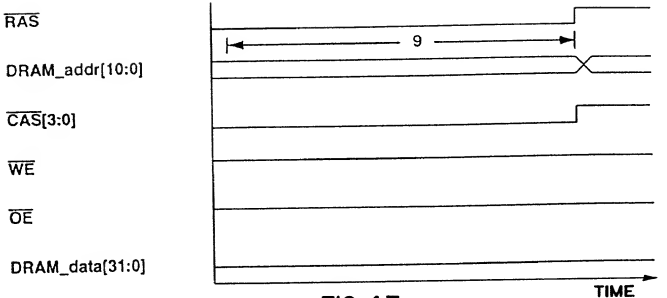

RAS
DRAM_addr[10:0]
CAS[3:0]
WE
OE
DRAM_data[31:0]
9
TIME

FIG.47

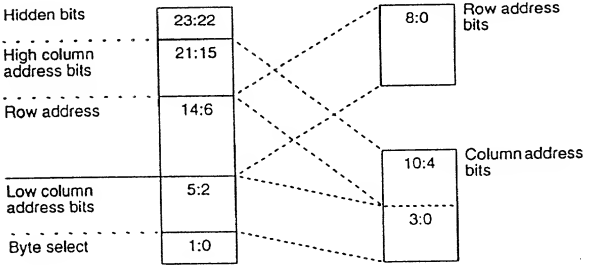

Hidden bits
High column address bits
Row address
Low column address bits
Byte select
23:22
21:15
14:6
5:2
1:0
8:0
Row address bits
10:4
3:0
Column address bits

FIG.48

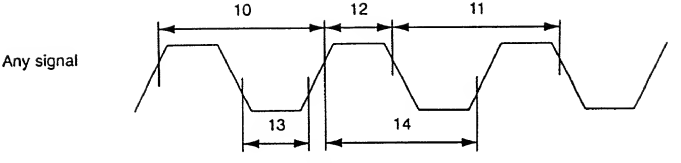


FIG.49

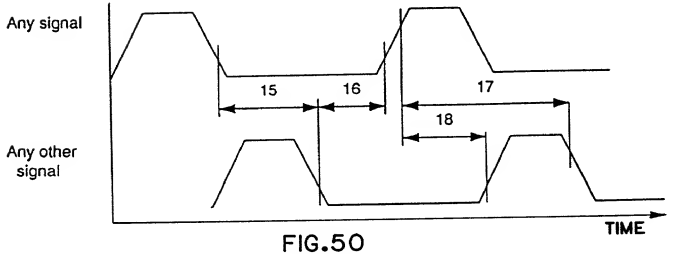


FIG.50

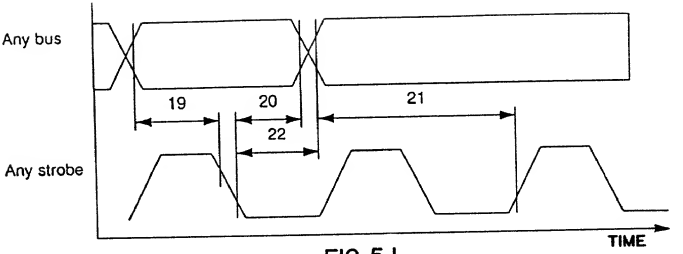


FIG.51

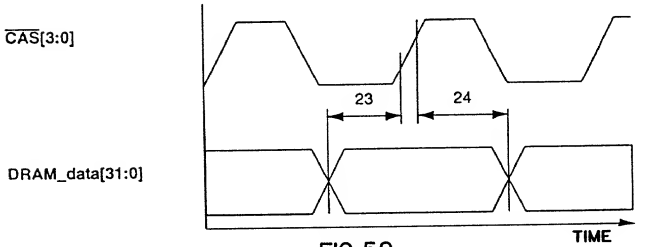


FIG.52

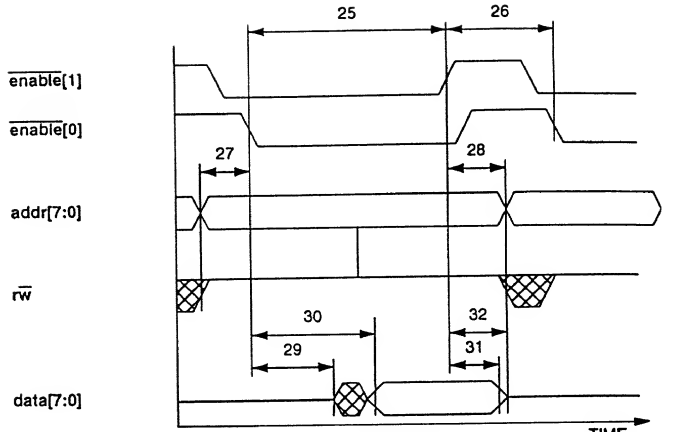


FIG.53

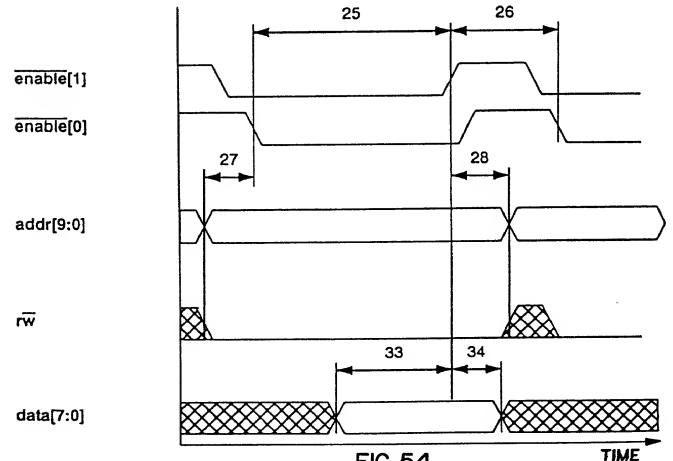


FIG.54

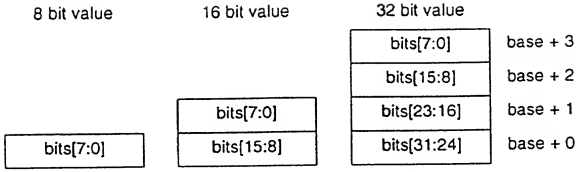


FIG.55

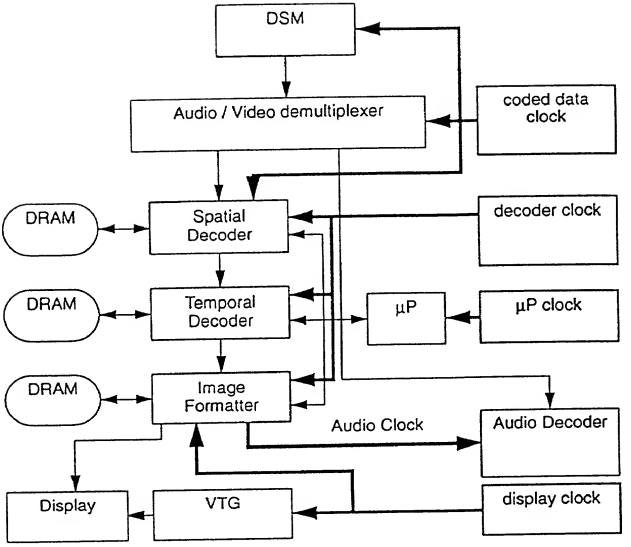


FIG.56

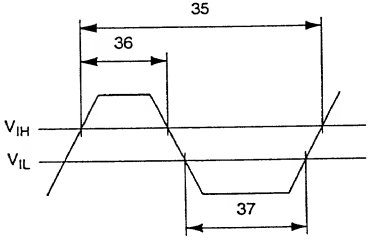


FIG.57

Off-chip DRAM
Coded data buffer
Token buffer
DRAM interface
Start-up control
Huffman decode and video demultiplex
Inverse modeler
Start code detection
Inverse quantiser
Input circuit
Inverse DCT
Coded video data
Microprocessor interface (MPI)
Error signal, motion vectors and other Tokens

FIG.58

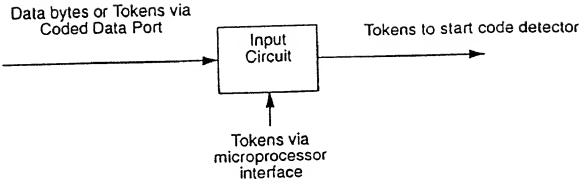


FIG.59

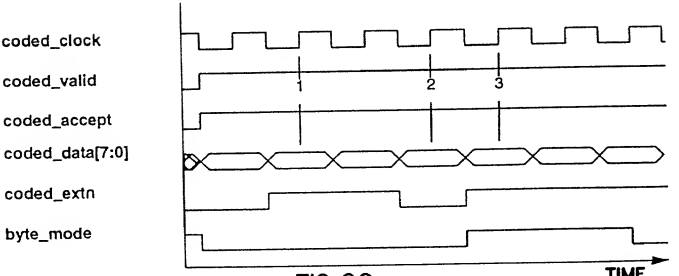


FIG.60

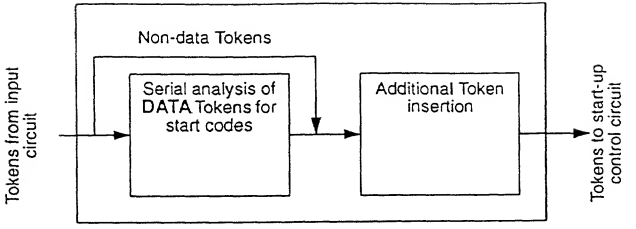


FIG.6 I

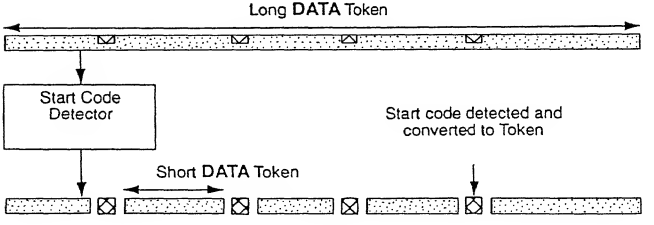


FIG.62

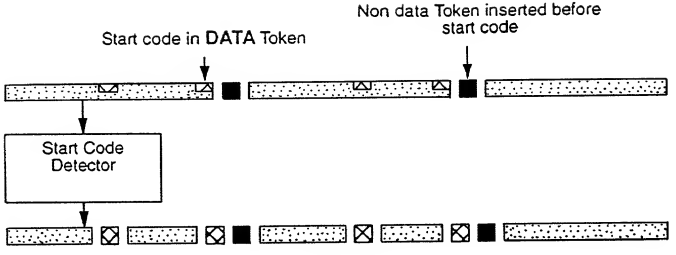

Non data Token inserted before start code
Start code in DATA Token
Start Code Detector

FIG.63

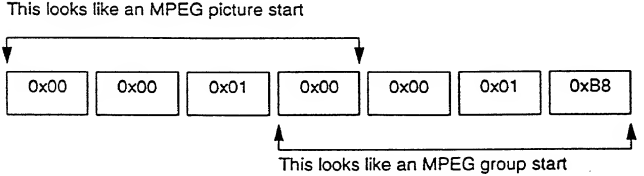

This looks like an MPEG picture start
0x00
0x00
0x01
0x00
0x00
0x01
0xB8
This looks like an MPEG group start

FIG.64

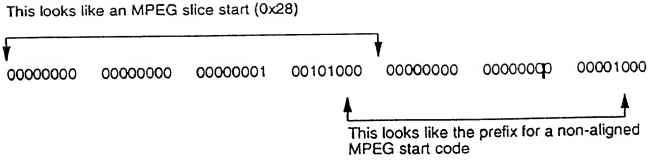

This looks like an MPEG slice start (0x28)
00000000 00000000 00000001 00101000 00000000 00000000 00001000
This looks like the prefix for a non-aligned MPEG start code

FIG.65

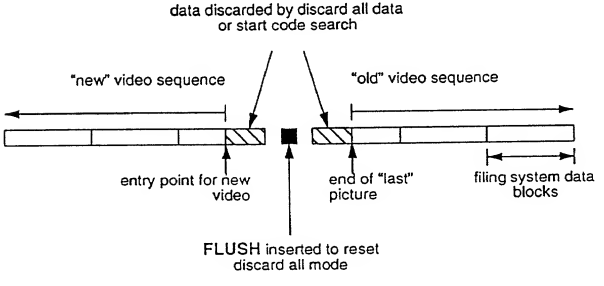

data discarded by discard all data or start code search
"new" video sequence
"old" video sequence
entry point for new video
end of "last" picture
filing system data blocks
FLUSH inserted to reset discard all mode

FIG.66

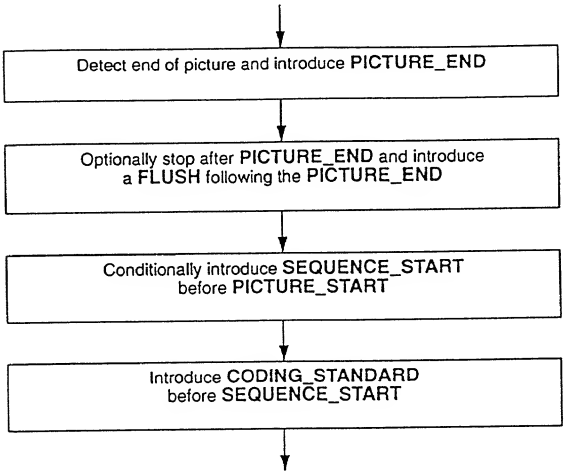


FIG.67

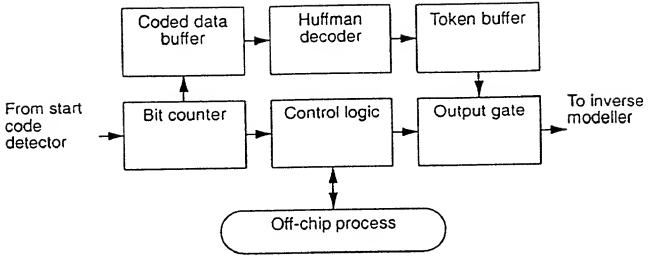

Coded data buffer
Huffman decoder
Token buffer
From start code detector
Bit counter
Control logic
Output gate
To inverse modeller
Off-chip process

FIG.68

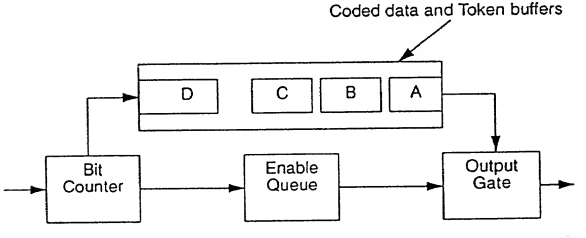

Coded data and Token buffers
D
C
B
A
Bit Counter
Enable Queue
Output Gate

FIG.69

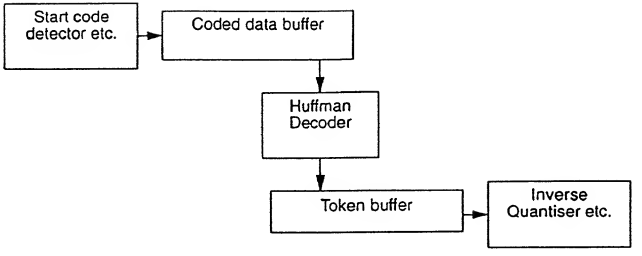

Start code detector etc.
Coded data buffer
Huffman Decoder
Token buffer
Inverse Quantiser etc.

FIG.70

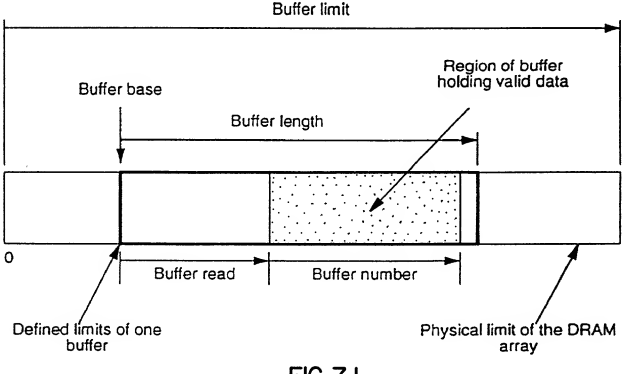

Buffer limit
Region of buffer holding valid data
Buffer base
Buffer length
0
Buffer read
Buffer number
Defined limits of one buffer
Physical limit of the DRAM array

FIG.71

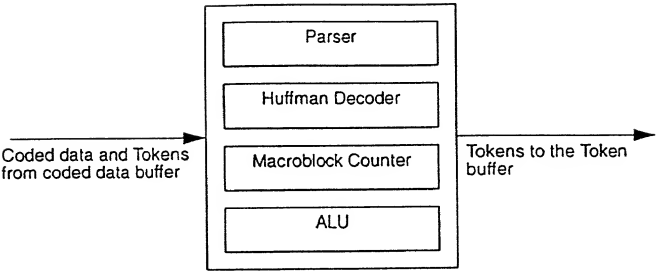


FIG.72

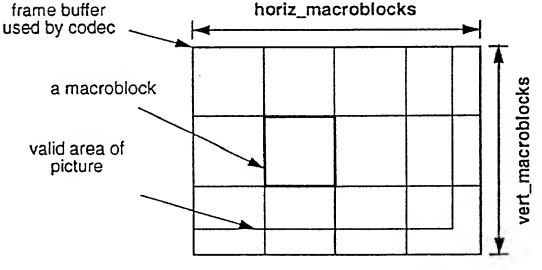


FIG.73

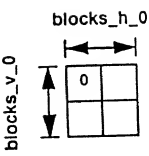


FIG.74A

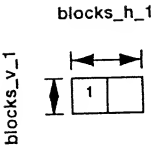


FIG.74B

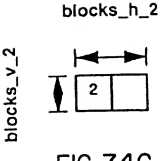


FIG.74C

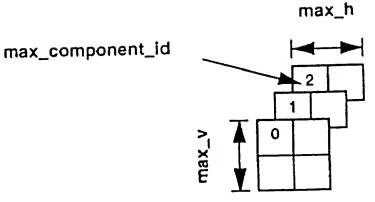


FIG.74D

$$\begin{cases} \text{horiz\_macroblocks} = \dfrac{\text{horiz\_pels} + 15}{16} \\ \text{vert\_macroblocks} = \dfrac{\text{vert\_pels} + 15}{16} \end{cases}$$

FIG.75

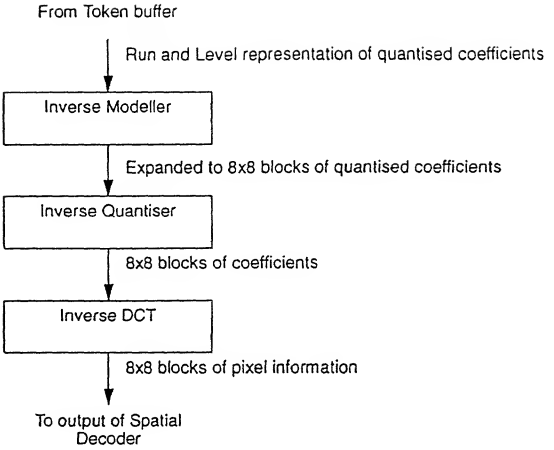

From Token buffer
Run and Level representation of quantised coefficients
Inverse Modeller
Expanded to 8x8 blocks of quantised coefficients
Inverse Quantiser
8x8 blocks of coefficients
Inverse DCT
8x8 blocks of pixel information
To output of Spatial Decoder

FIG.76

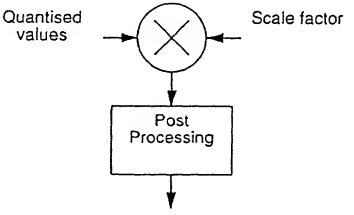

Quantised values
Scale factor
Post Processing

FIG.77

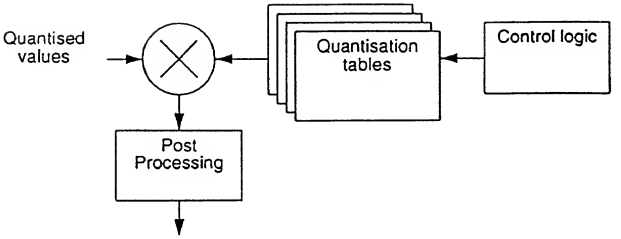


FIG.78

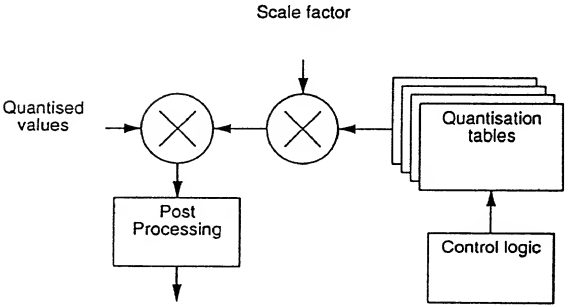


FIG.79

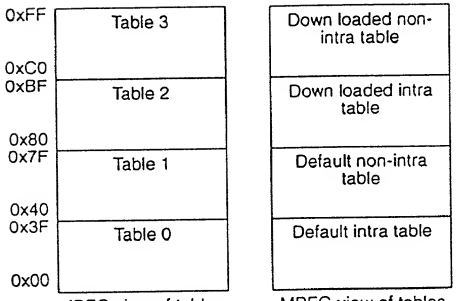


FIG.80

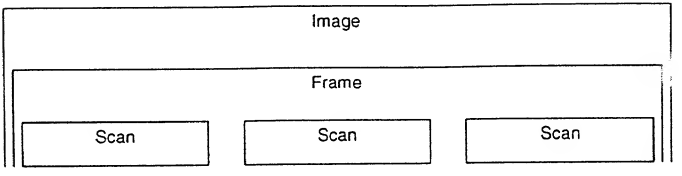


FIG.81

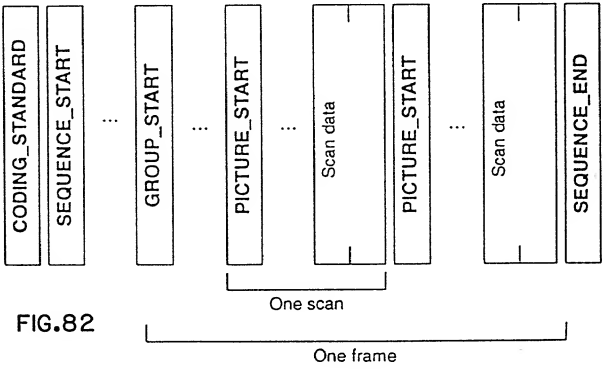


FIG.82

FIG.83

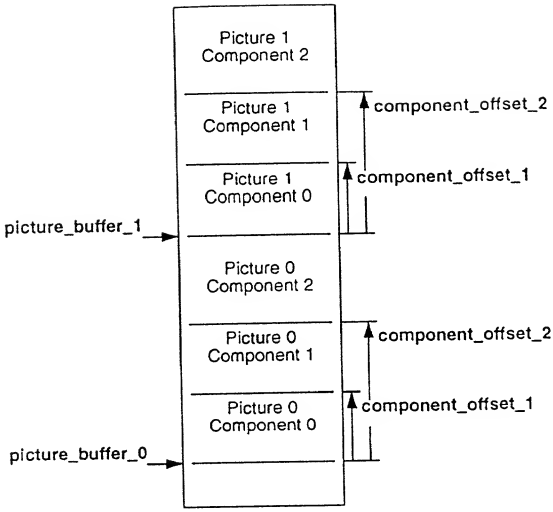


FIG.84

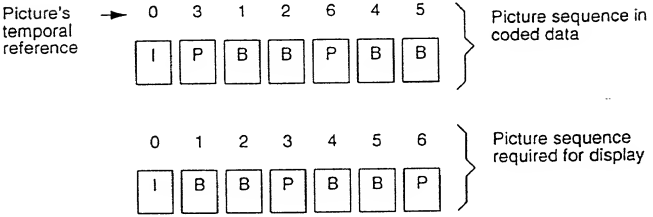


FIG.85

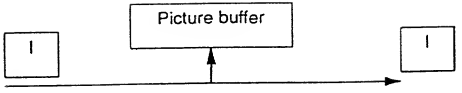


FIG.86

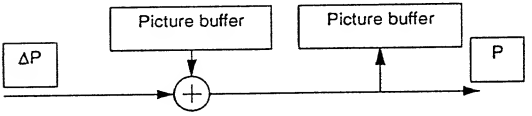


FIG.87

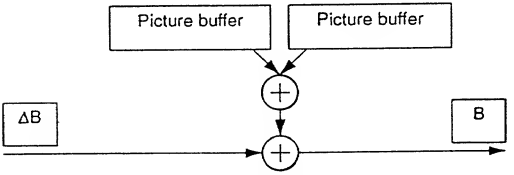


FIG.88

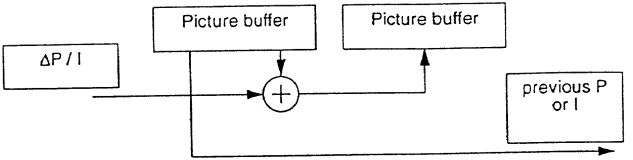

Picture buffer
Picture buffer
ΔP / I
previous P
or I

FIG.89

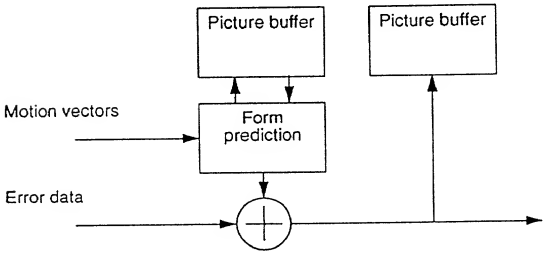

Picture buffer
Picture buffer
Motion vectors
Form
prediction
Error data

FIG.90

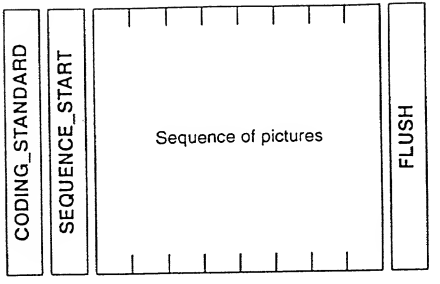

CODING_STANDARD
SEQUENCE_START
Sequence of pictures
FLUSH

FIG.9 I

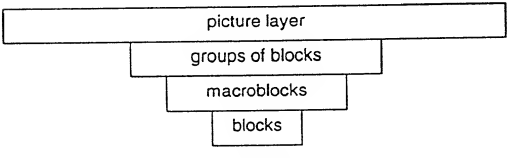

picture layer
groups of blocks
macroblocks
blocks

FIG.92

FIG.93

FIG.94

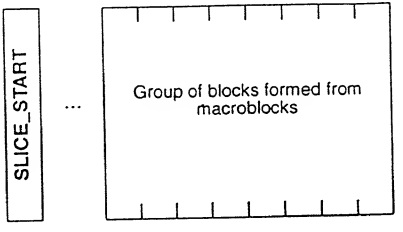


FIG.95

1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22
23	24	25	26	27	28	29	30	31	32	33

FIG.96

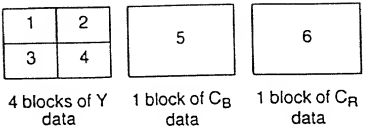


FIG.97

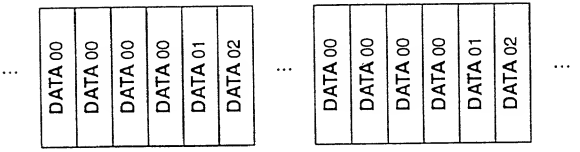


FIG.98

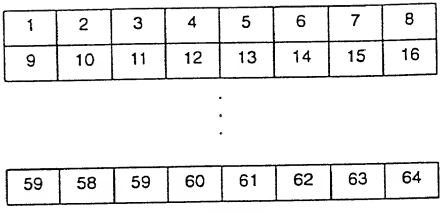


FIG.99

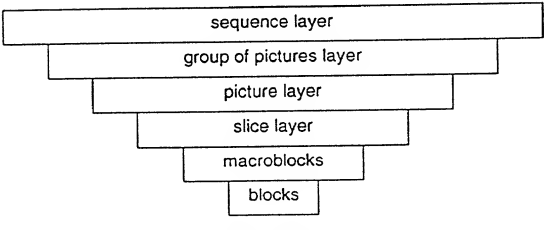


FIG.100

FIG. 101

FIG. 102

FIG. 103

FIG. 104

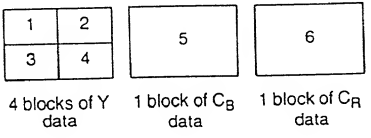


FIG. 105

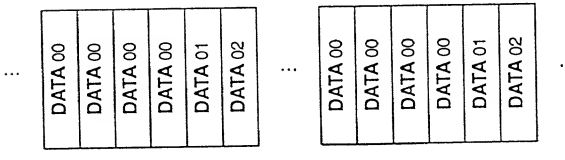


FIG. 106

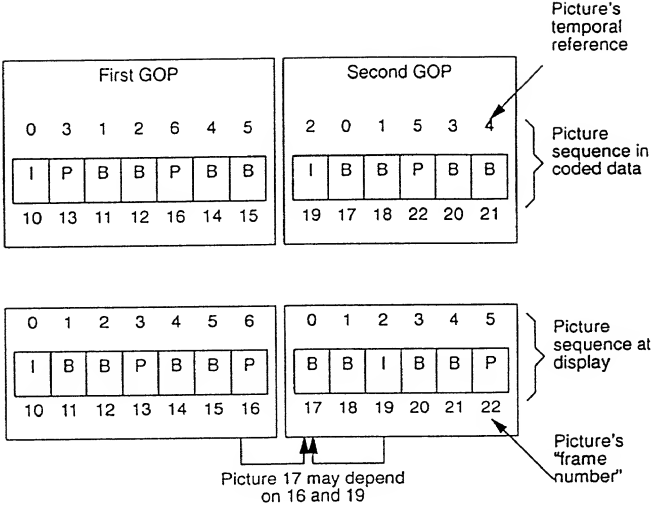


FIG. 107

FIG. 108

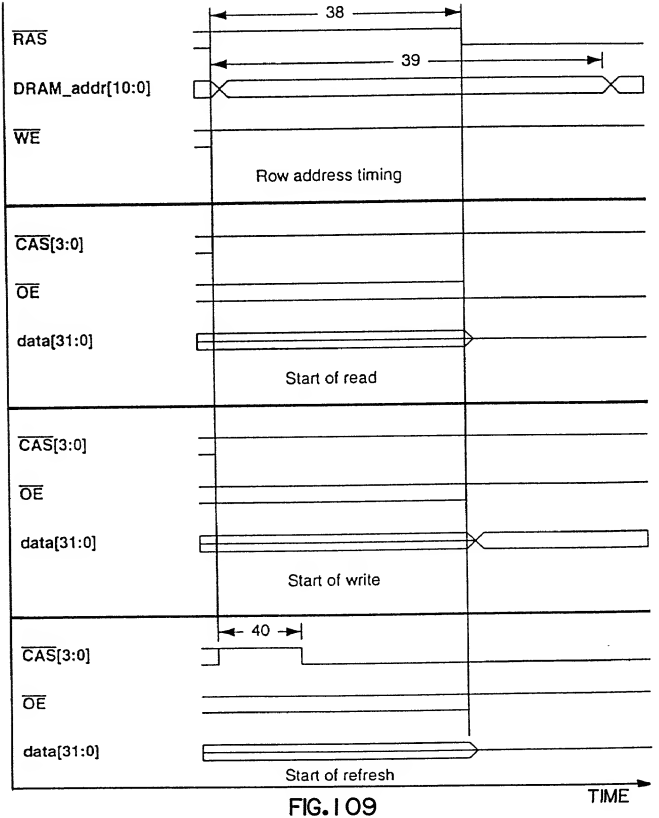


FIG. 109

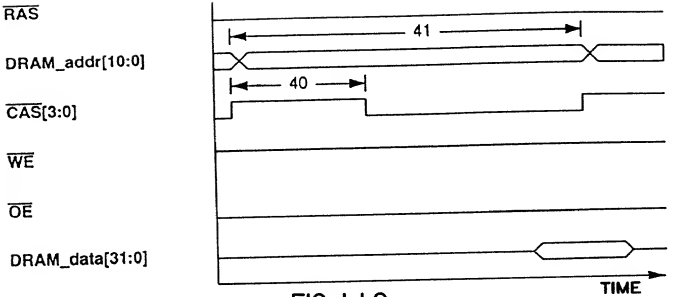


FIG. 1 1 0

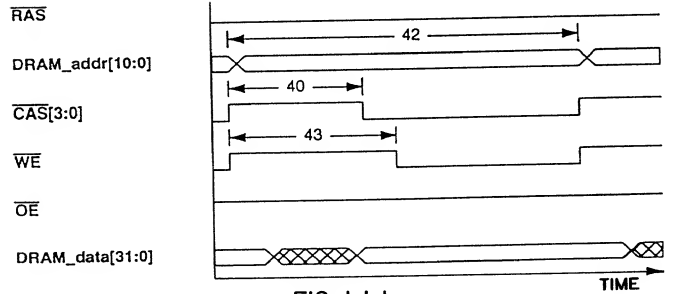


FIG. 1 1 1

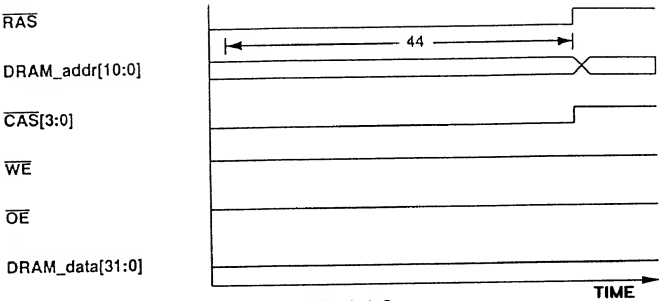


FIG. 112

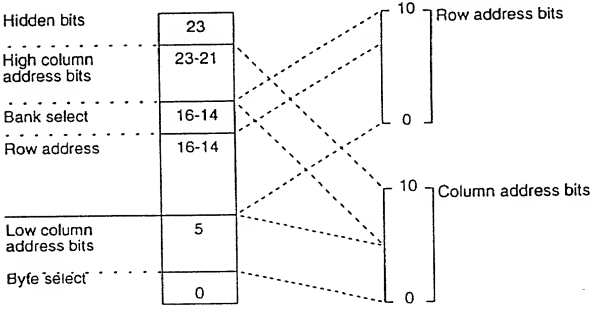


FIG. 113

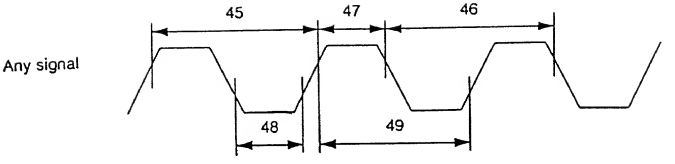


FIG. 114

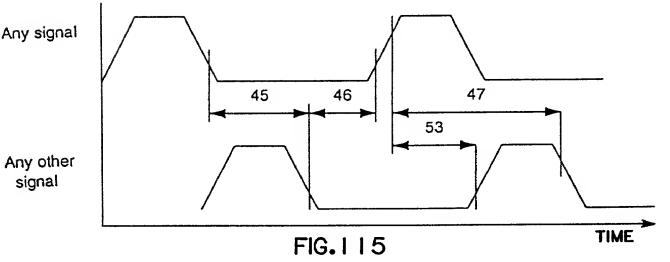


FIG. 115

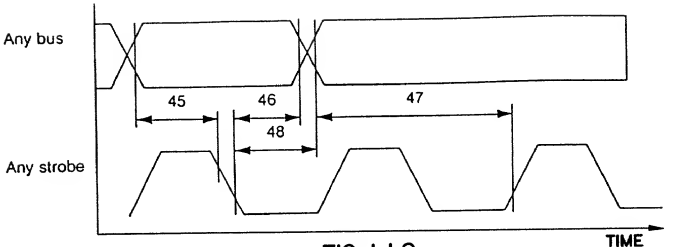


FIG. 116

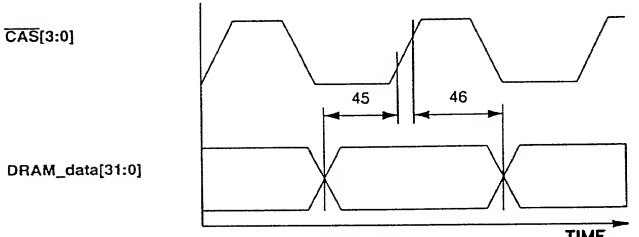


FIG. 117

FIG. 118

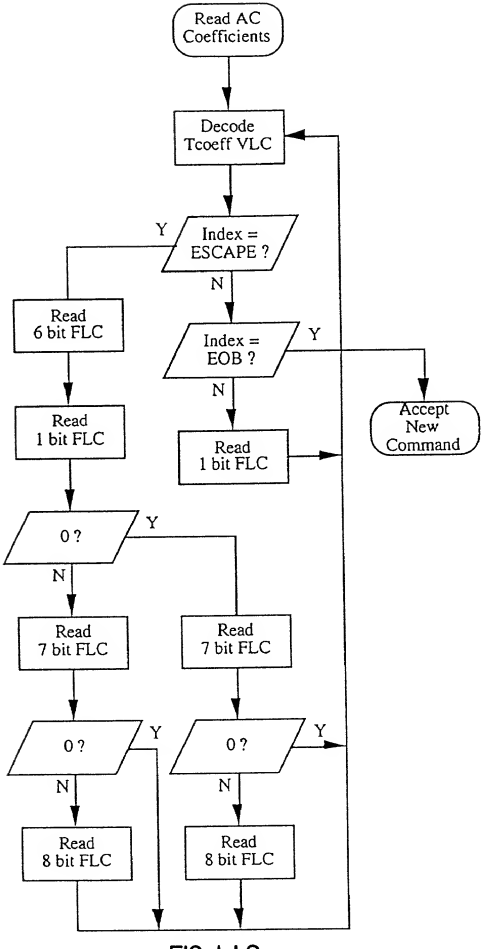


FIG. 1 1 9

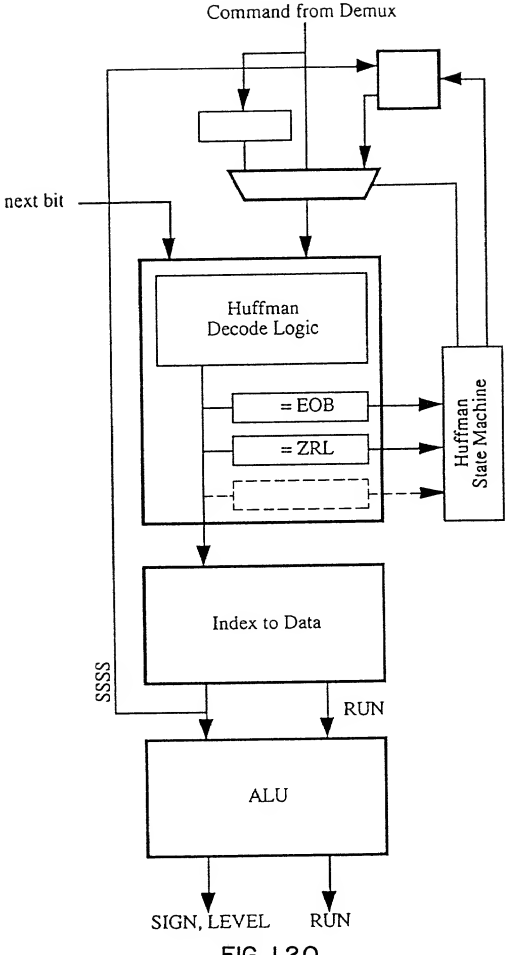


FIG. 120

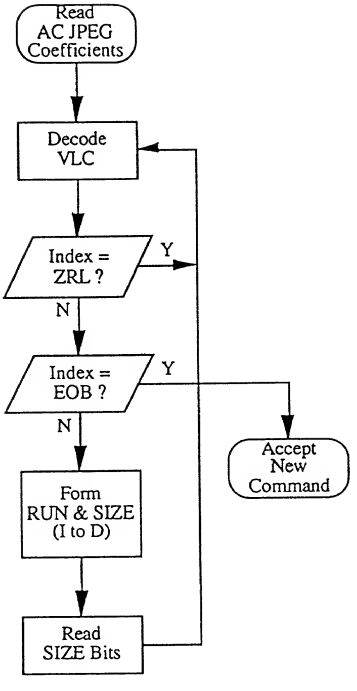


FIG. 12 1A

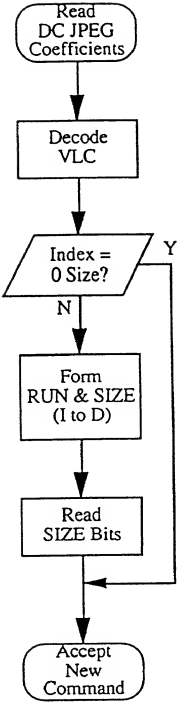


FIG. 12 1B

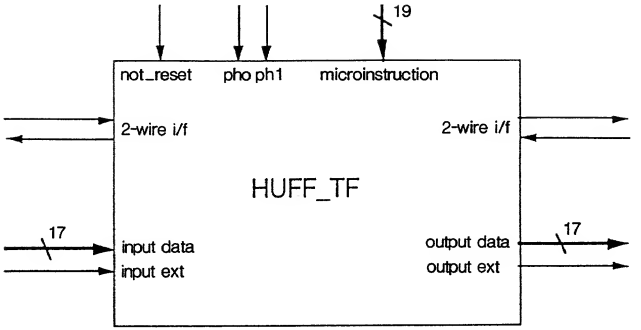


FIG. 122

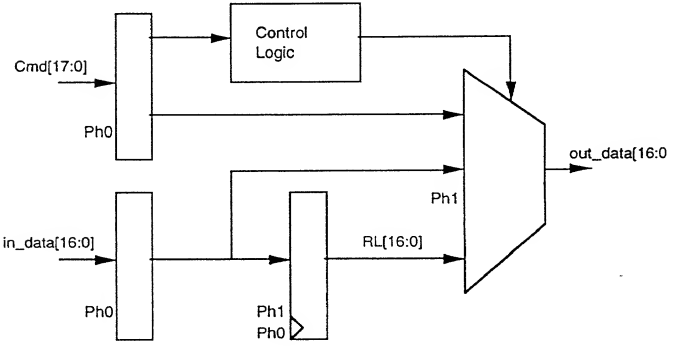


FIG. 123

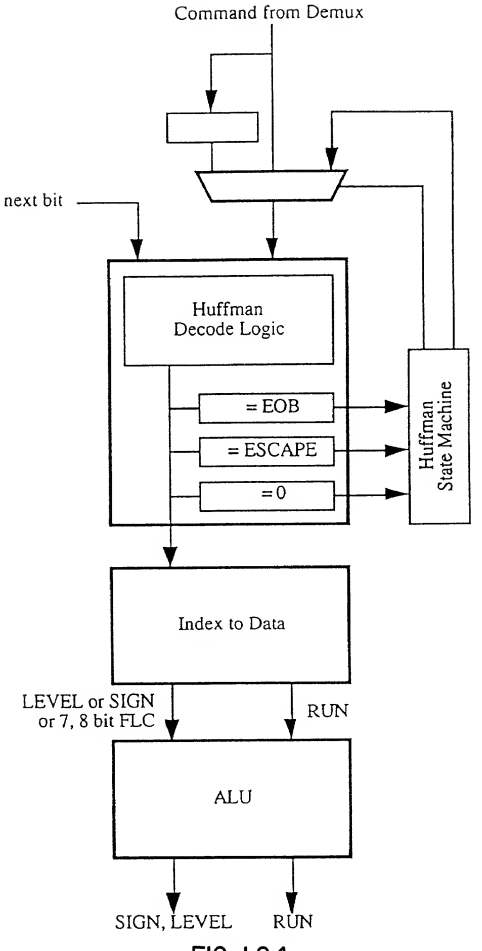

Command from Demux
next bit
Huffman
Decode Logic
= EOB
= ESCAPE
= 0
Huffman
State Machine
Index to Data
LEVEL or SIGN
or 7, 8 bit FLC
RUN
ALU
SIGN, LEVEL
RUN

FIG. 124

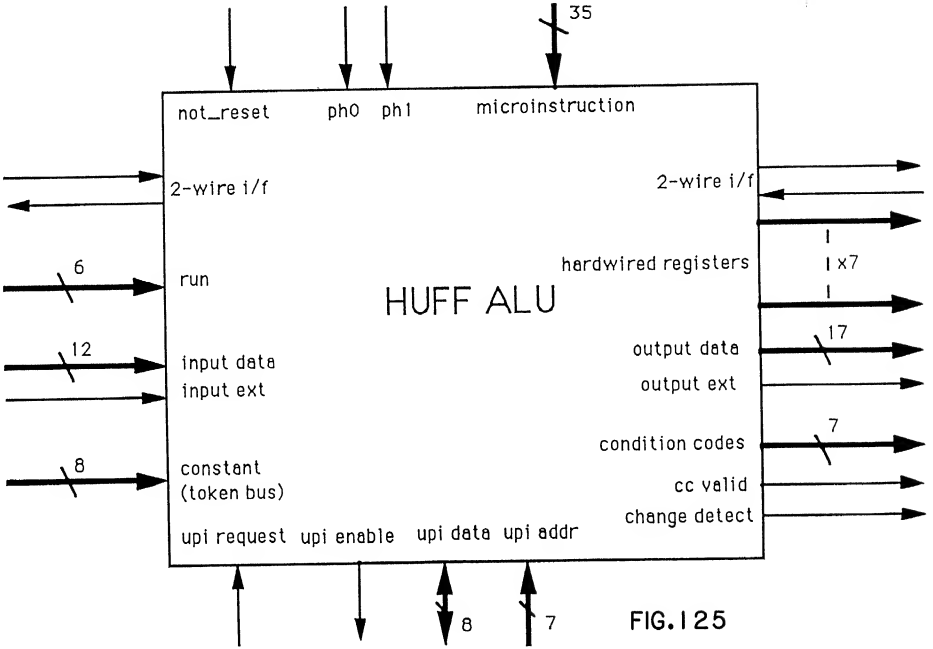


FIG. 125

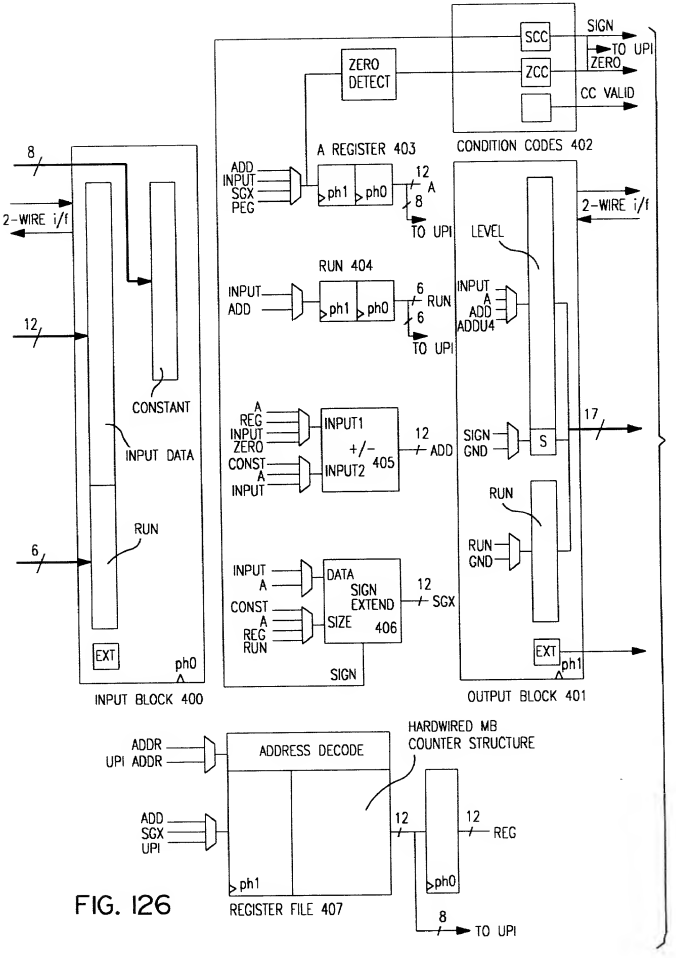

SCC
SIGN
TO UPI
ZERO DETECT
ZCC
ZERO
CC VALID
CONDITION CODES 402
A REGISTER 403
ADD
INPUT
SGX
PEG
ph1
ph0
12
A
8
TO UPI
8
2-WIRE i/f
LEVEL
RUN 404
INPUT
ADD
6
RUN
INPUT
A
ADD
ADDU4
12
CONSTANT
INPUT DATA
A
REG
INPUT
ZERO
CONST
A
INPUT
INPUT1
+/-
INPUT2
405
12
ADD
SIGN
GND
S
17
RUN
6
RUN
RUN
GND
INPUT
A
CONST
A
REG
RUN
DATA
SIGN
EXTEND
SIZE
406
12
SGX
SIGN
EXT
ph0
ph1
INPUT BLOCK 400
OUTPUT BLOCK 401
HARDWIRED MB
COUNTER STRUCTURE
ADDR
UPI ADDR
ADDRESS DECODE
ADD
SGX
UPI
12
12
REG
ph1
ph0
8
TO UPI
REGISTER FILE 407

FIG. 126

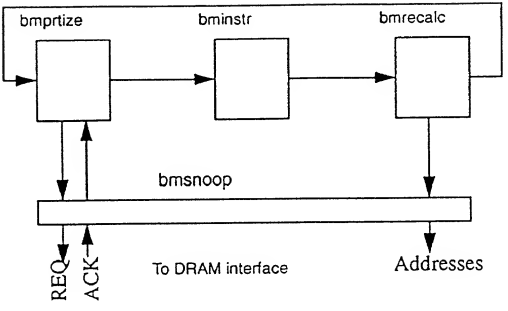


FIG. 127

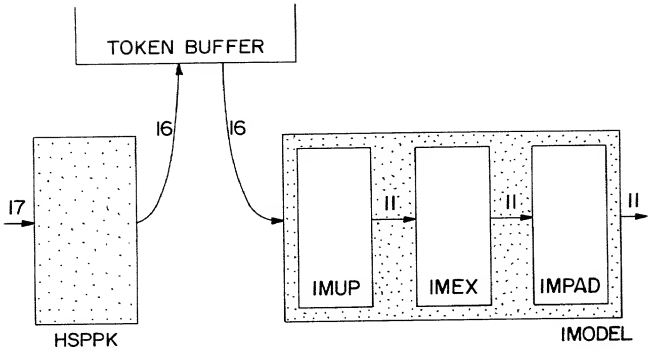


FIG. 128

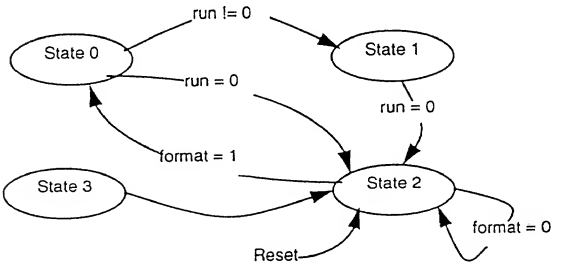


FIG. 129

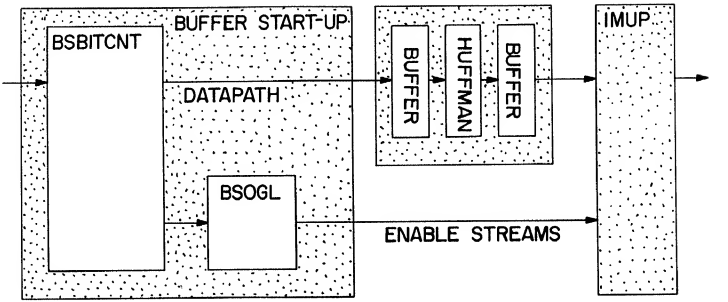


FIG. 130

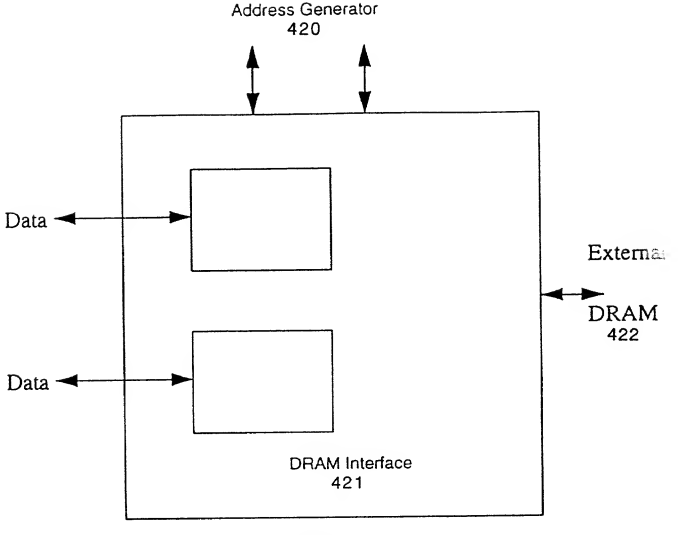


FIG. 131

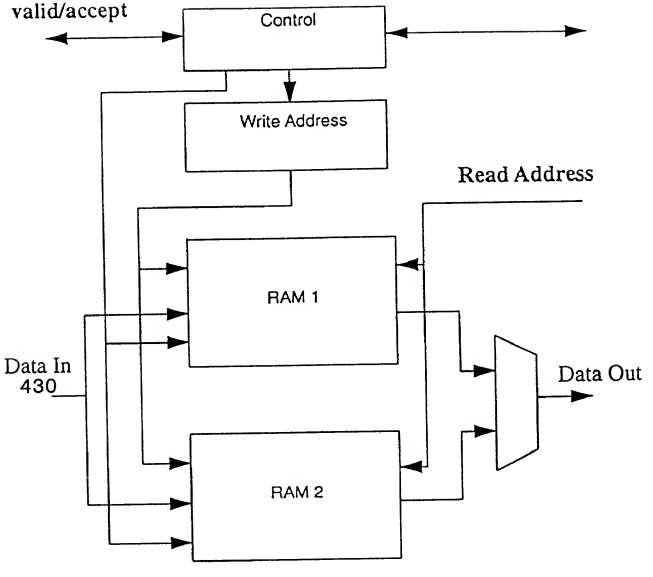


FIG. 132

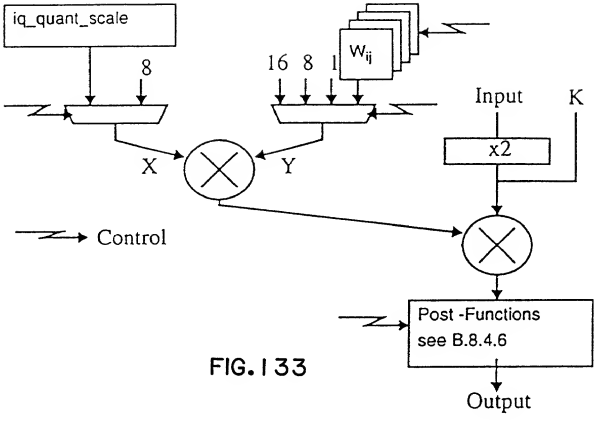


FIG. 133

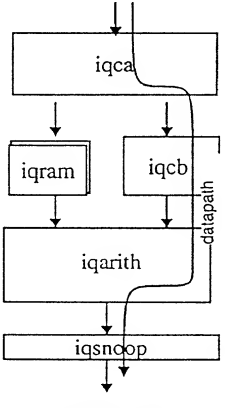


FIG. 134

FIG. 135

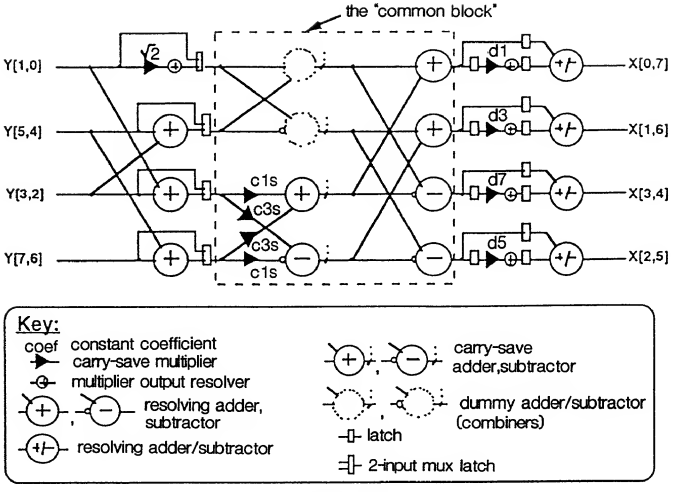


FIG. 137

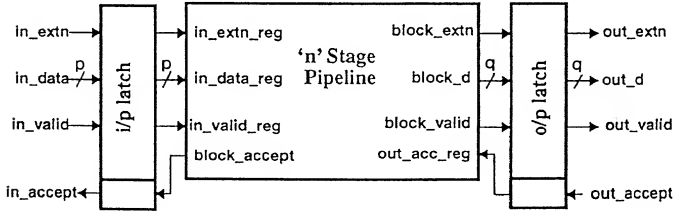


FIG. 138

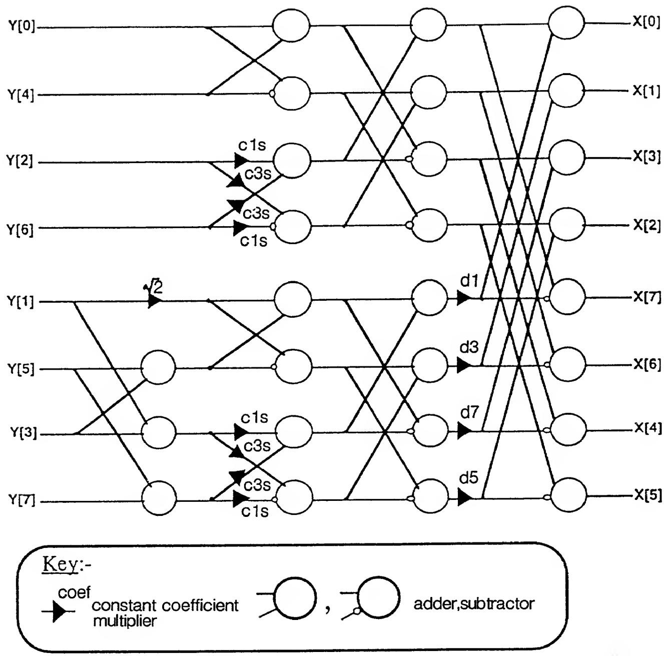

Y[0]
Y[4]
Y[2]
Y[6]
Y[1]
Y[5]
Y[3]
Y[7]
√2
c1s
c3s
c3s
c1s
c1s
c3s
c3s
c1s
d1
d3
d7
d5
X[0]
X[1]
X[3]
X[2]
X[7]
X[6]
X[4]
X[5]
Key:-
coef
constant coefficient multiplier
adder,subtractor

FIG. 136

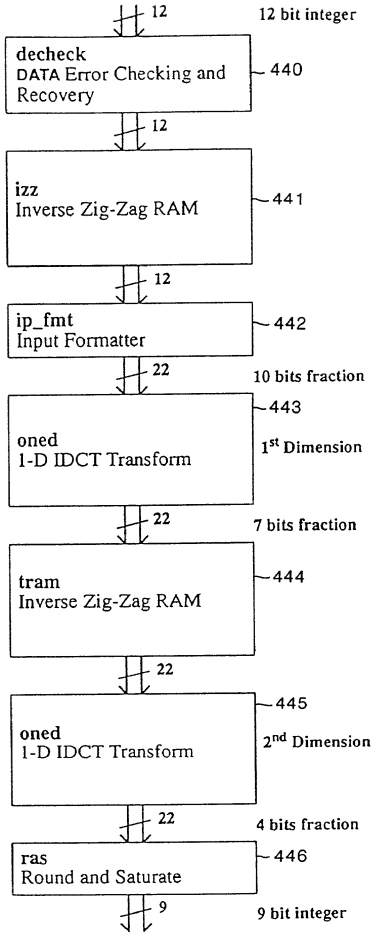

12
12 bit integer
decheck
DATA Error Checking and Recovery
440
12
izz
Inverse Zig-Zag RAM
441
12
ip_fmt
Input Formatter
442
22
10 bits fraction
443
oned
1-D IDCT Transform
1st Dimension
22
7 bits fraction
tram
Inverse Zig-Zag RAM
444
22
445
oned
1-D IDCT Transform
2nd Dimension
22
4 bits fraction
ras
Round and Saturate
446
9
9 bit integer

FIG. 139

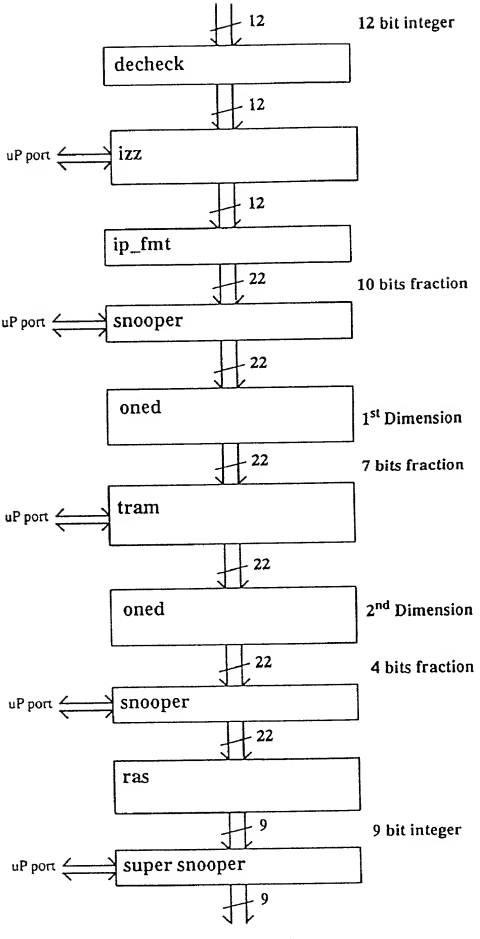


FIG. 140

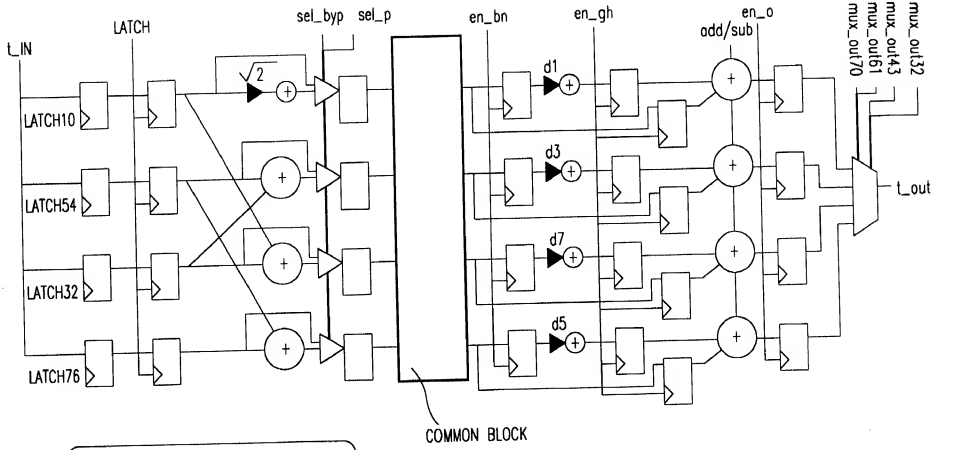


FIG. 141

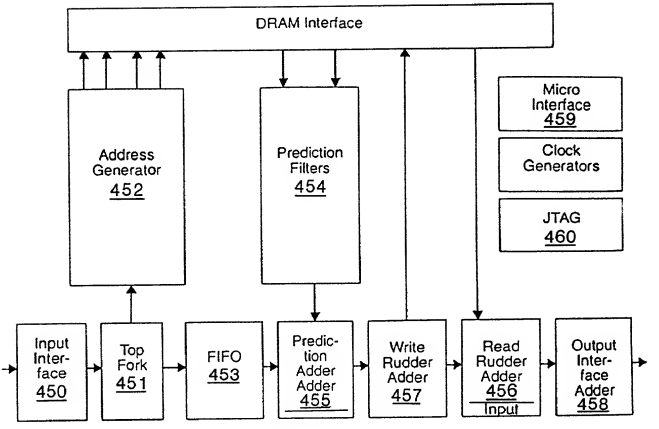


FIG. 142

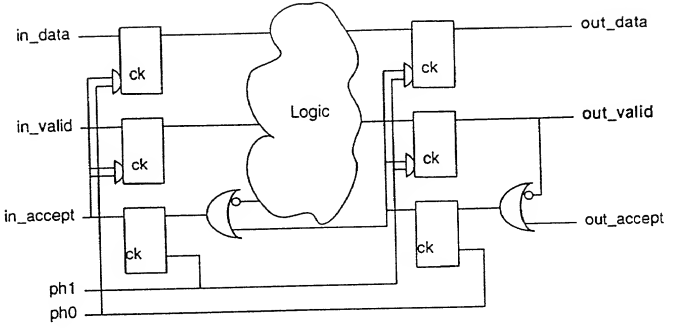

in_data
in_valid
in_accept
ph1
ph0
ck
Logic
out_data
out_valid
out_accept

FIG. 143

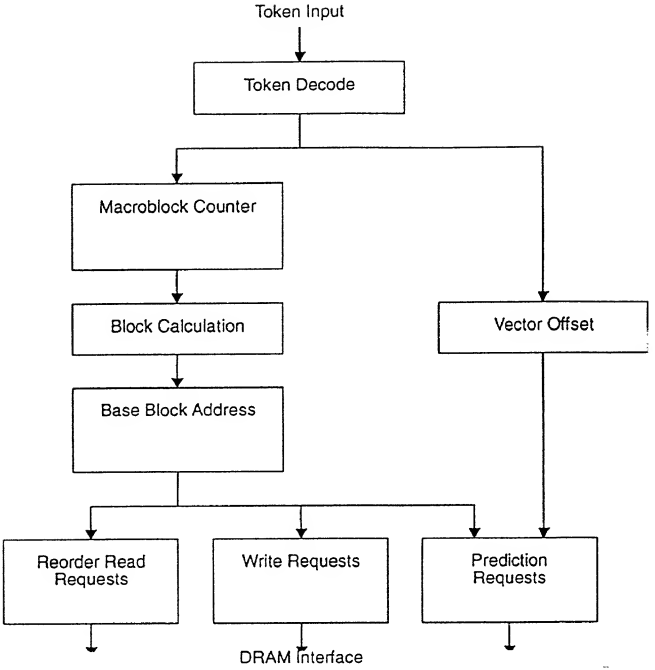


FIG. 144

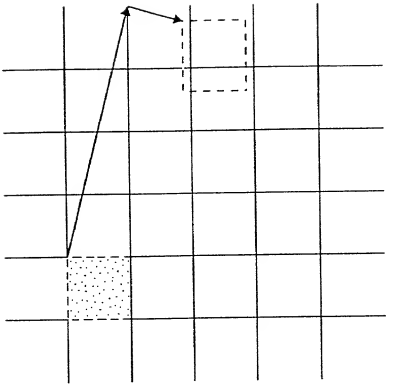


FIG. 145

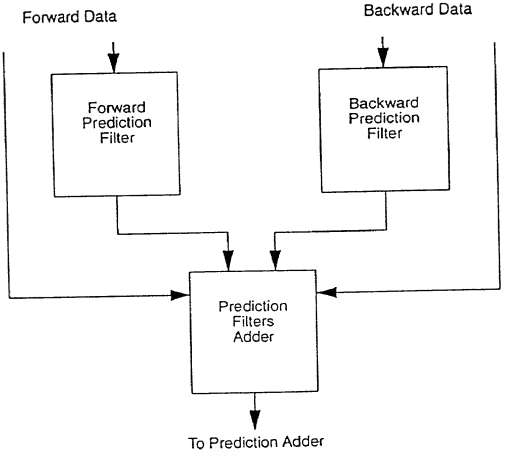


FIG. 146

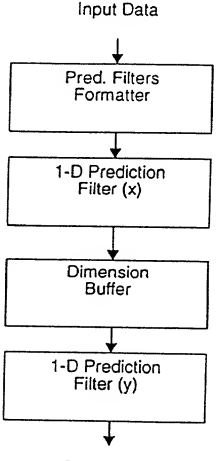

Input Data
Pred. Filters Formatter
1-D Prediction Filter (x)
Dimension Buffer
1-D Prediction Filter (y)
Output Data

FIG. 147

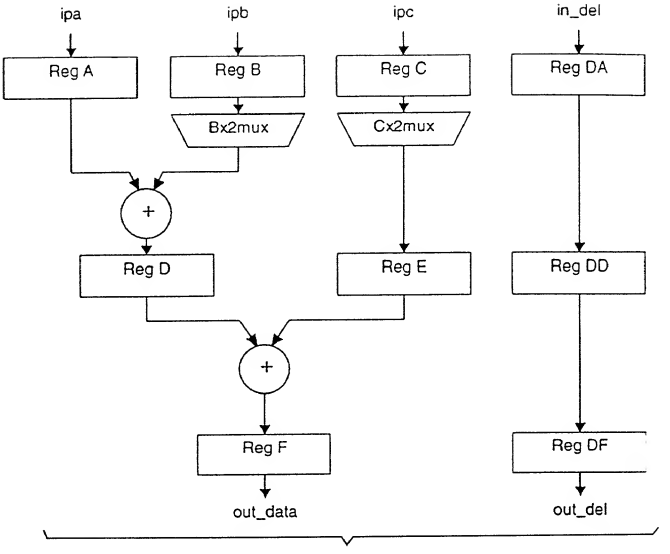


FIG. 148

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

FIG. 149

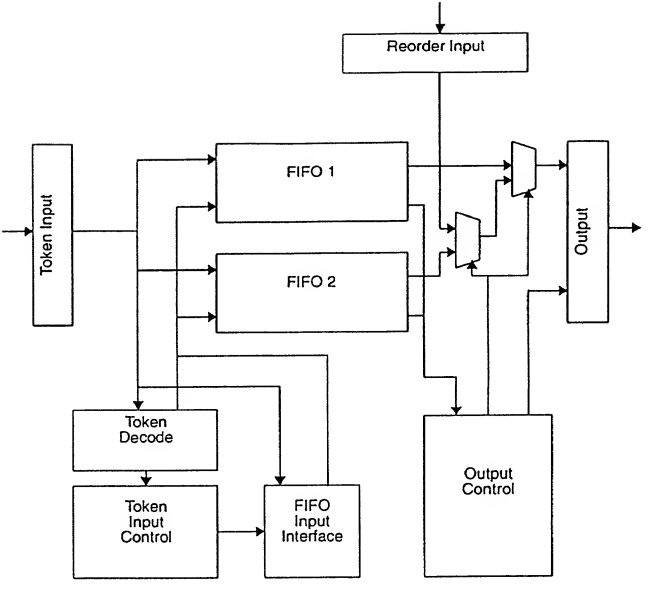


FIG. 150

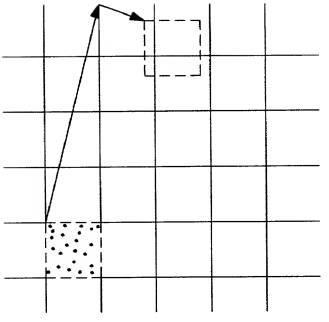


FIG. 151

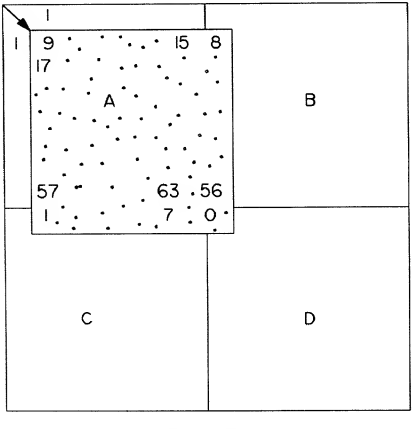


FIG. 152

Read Cycle

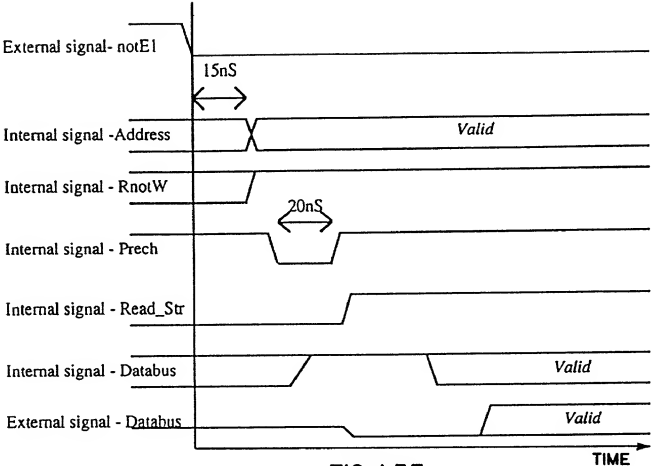


FIG. 153

Write Cycle

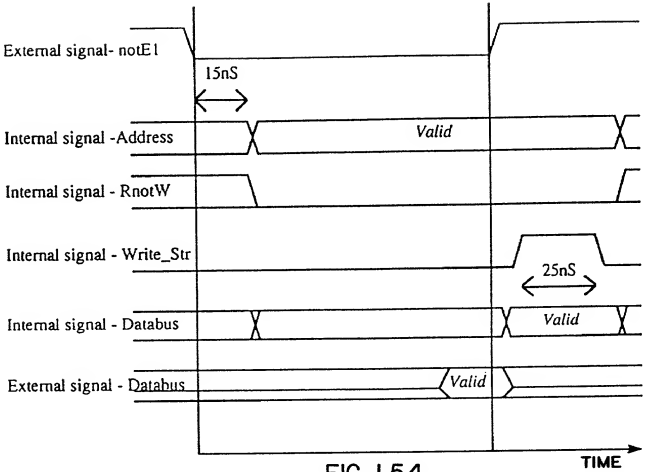


FIG. 154

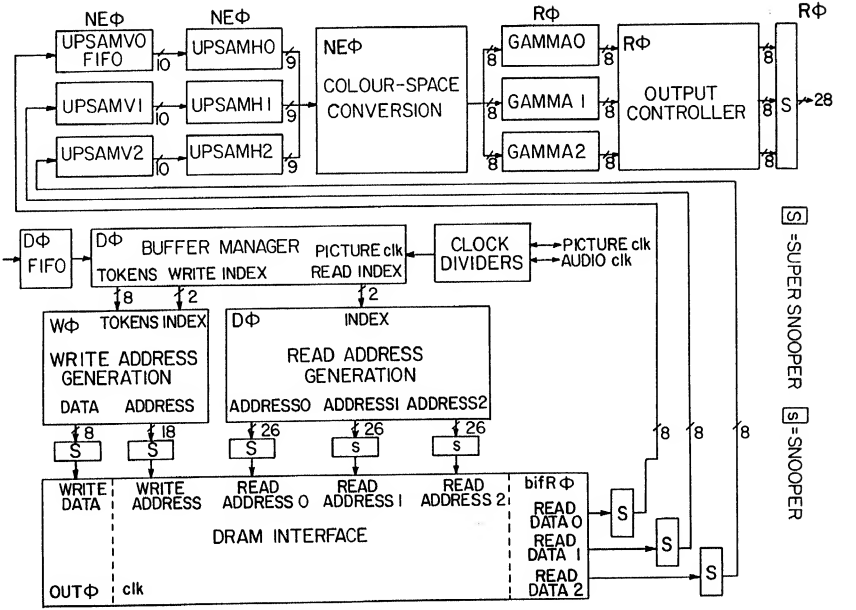


FIG. 155

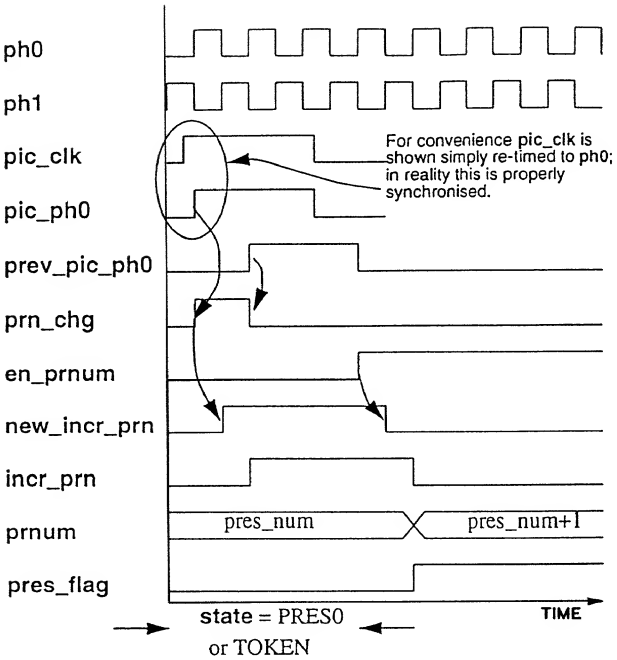

ph0
ph1
pic_clk
pic_ph0
prev_pic_ph0
prn_chg
en_prnum
new_incr_prn
incr_prn
prnum
pres_flag
For convenience pic_clk is shown simply re-timed to ph0; in reality this is properly synchronised.
pres_num
pres_num+1
state = PRES0
or TOKEN
TIME

FIG. 156

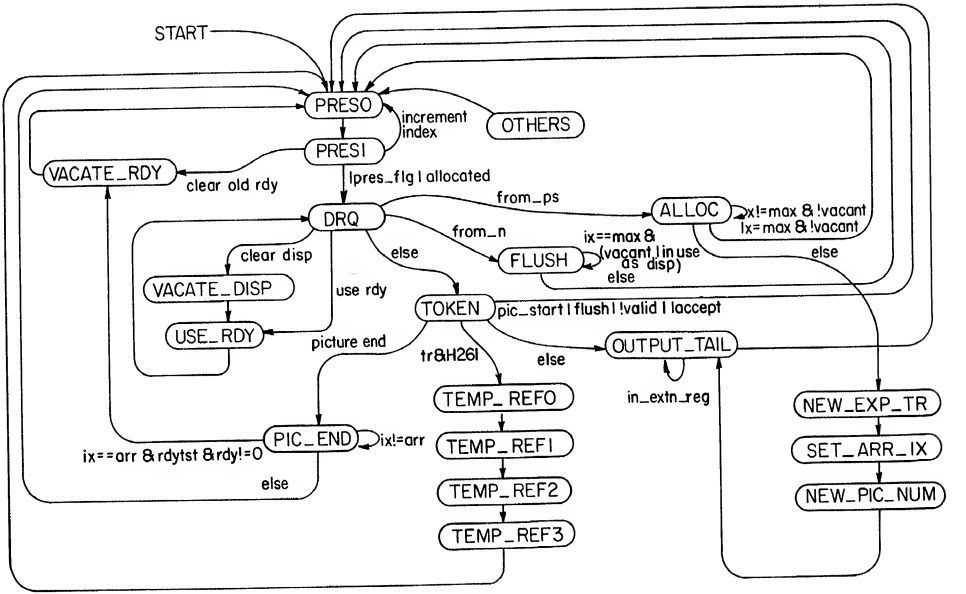


FIG. 157

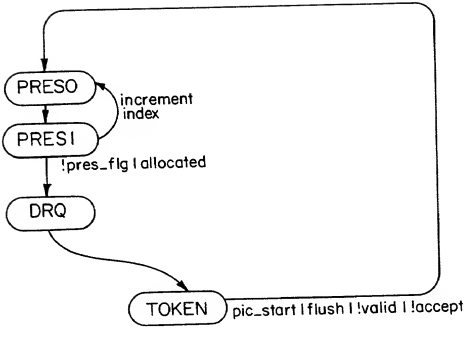

PRESO
increment
index
PRES1
!pres_flg | allocated
DRQ
TOKEN
pic_start | flush | !valid | !accept

FIG. 158

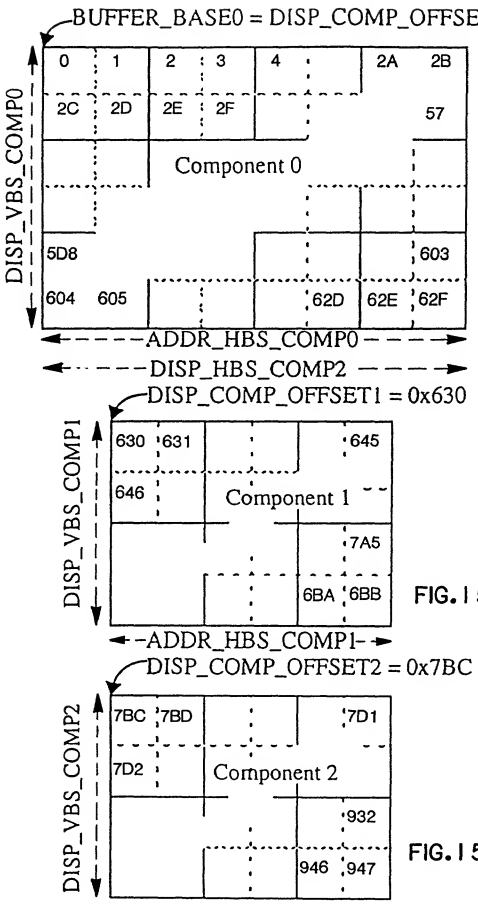


FIG. 159A

FIG. 159B

FIG. 159C

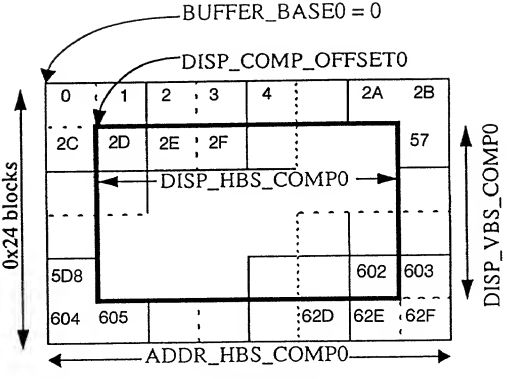


FIG. 160

BUFFER OFFSET 0x00

COMPONENT OFFSET 0x000 + ...

00	01	02	03	04	05	06	07	08	09	0A	0B
0C	0D	0E	0F	10	11	12	13	14	15	16	17
18	19	1A	1B	1C	1D	1E	1F	20	21	22	23
24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
30	31	32	33	34	35	36	37	38	39	3A	3B
3C	3D	3E	3F	40	41	42	43	44	45	46	47
48	49	4A	4B	4C	4D	4E	4F	50	51	52	53
54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67	68	69	6A	6B
6C	6D	6E	6F	70	71	72	73	74	75	76	77
78	79	7A	7B	7C	7D	7E	7F	80	81	82	83
84	85	86	87	88	89	8A	8B	8C	8D	8E	8F

FIG. 16 1A

COMPONENT1 OFFSET 0x100 + ...

00	01	02	03	04	05
06	07	08	09	0A	0B
0C	0D	0E	0F	10	11
12	13	14	15	16	17
18	19	1A	1B	1C	1D
1E	1F	20	21	22	23

FIG. 16 1B

COMPONENT1 OFFSET 0x200 + ...

00	01	02	03	04	05
06	07	08	09	0A	0B
0C	0D	0E	0F	10	11
12	13	14	15	16	17
18	19	1A	1B	1C	1D
1E	1F	20	21	22	23

FIG. 16 1C

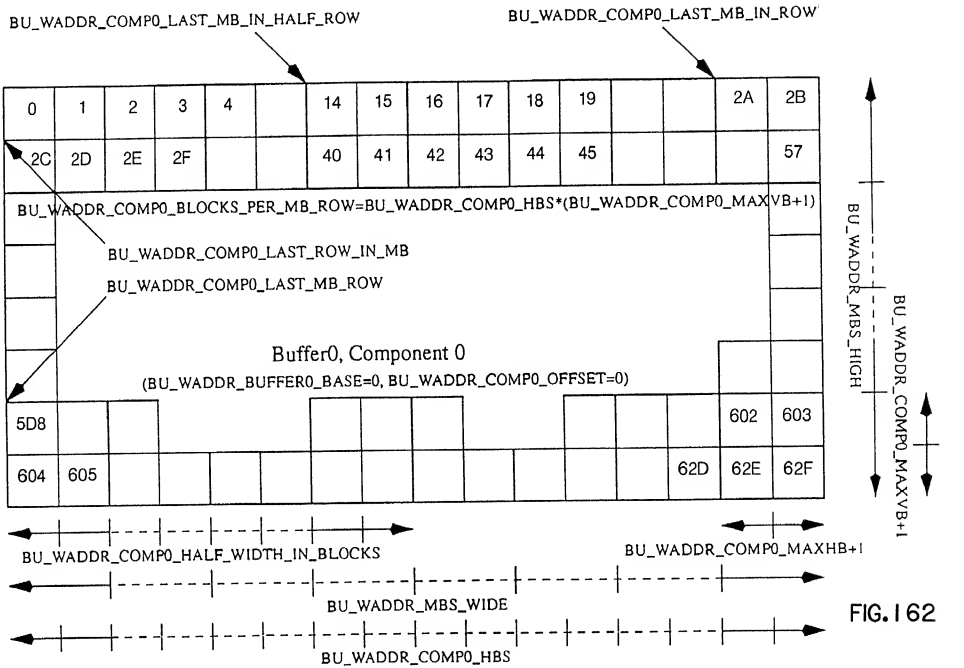
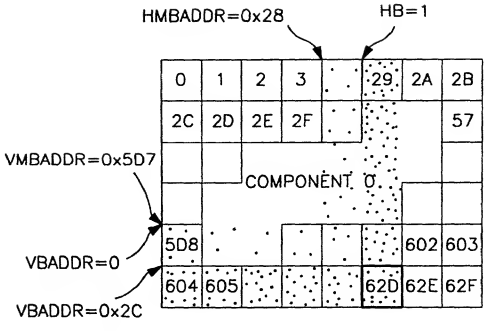
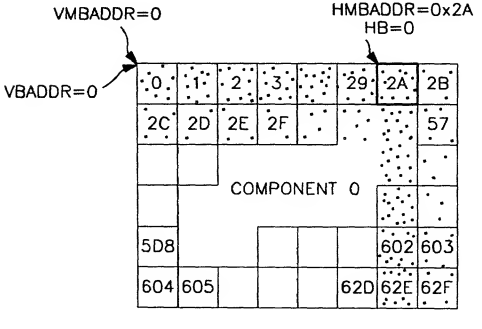


FIG. 162



BLOCK ADDRESS=0+0+0x5D8+0x28+0x2C+1=0x62D

FIG. 163A



BLOCK ADDRESS=0+0+0+0x2A+0+0=0x2A

FIG. 163B

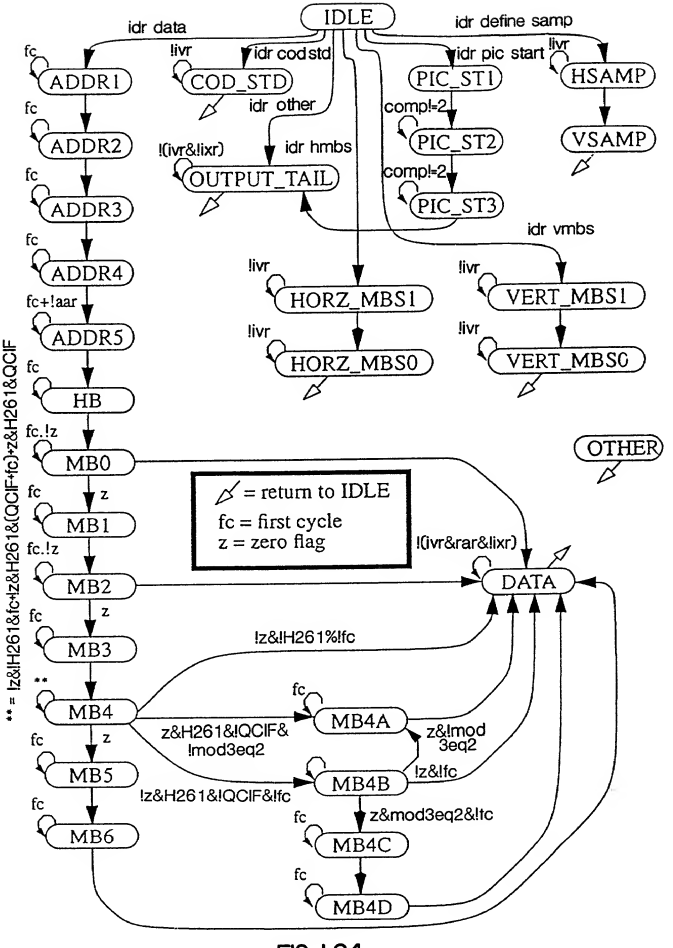

IDLE
idr data
idr define samp
idr cod std
idr pic start
idr other
idr hmbs
idr vmbs
ADDR1
ADDR2
ADDR3
ADDR4
ADDR5
HB
MB0
MB1
MB2
MB3
MB4
MB5
MB6
COD_STD
OUTPUT_TAIL
PIC_ST1
PIC_ST2
PIC_ST3
HSAMP
VSAMP
HORZ_MBS1
HORZ_MBS0
VERT_MBS1
VERT_MBS0
MB4A
MB4B
MB4C
MB4D
DATA
OTHER
fc
!ivr
!(ivr&!ixr)
fc+!aar
fc,!z
z
!z
comp!=2
= return to IDLE
fc = first cycle
z = zero flag
!(ivr&rar&!ixr)
!z&!H261%!fc
z&H261&!QCIF&
!mod3eq2
z&!mod
3eq2
!z&!fc
!z&H261&!QCIF&!fc
z&mod3eq2&!fc
** = !z&!H261&!fc+!z&H261&(QCIF+fc)+z&H261&QCIF

FIG. 164

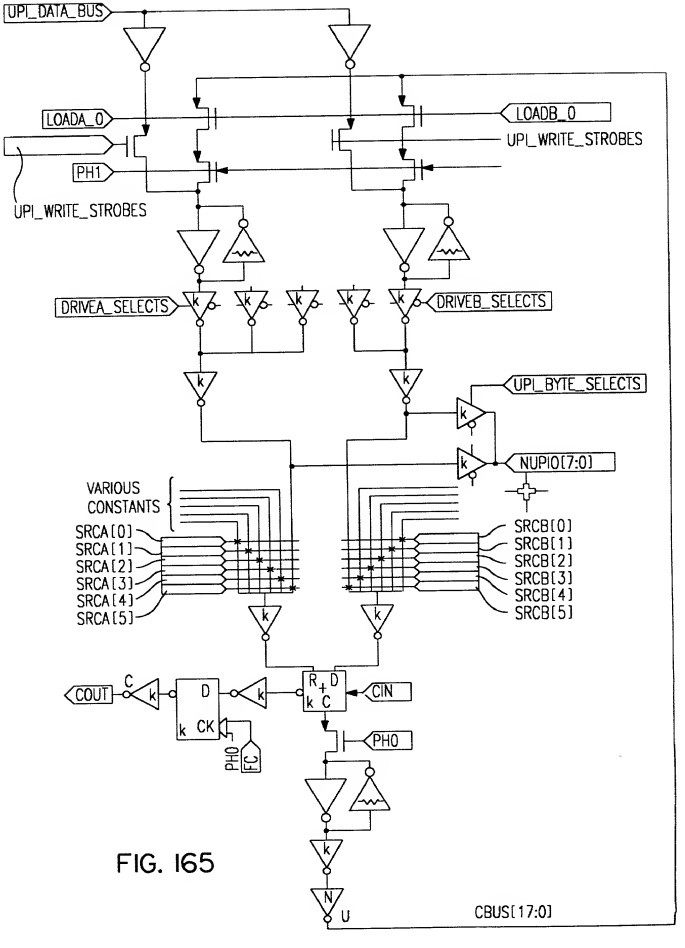


FIG. 165

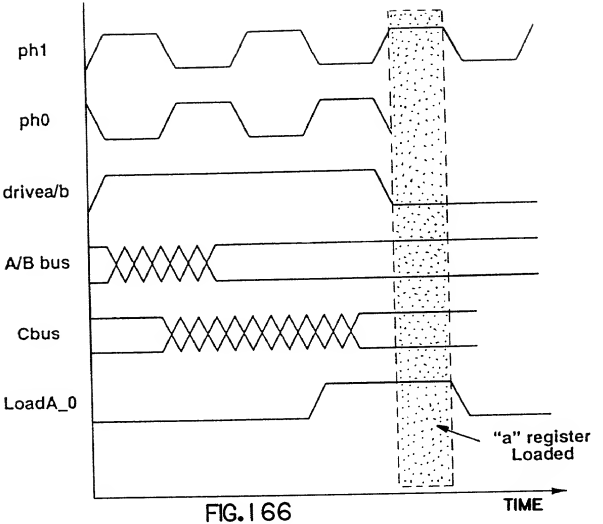


FIG. 166

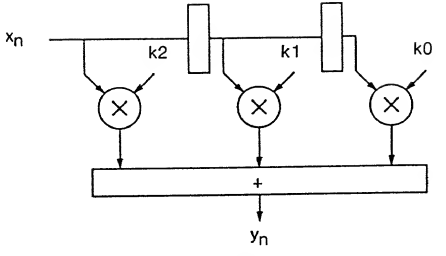


FIG. 167

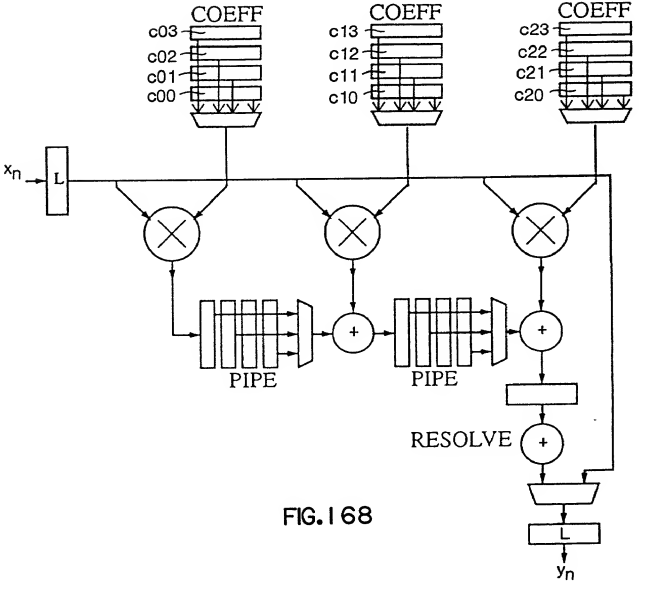


FIG. 168

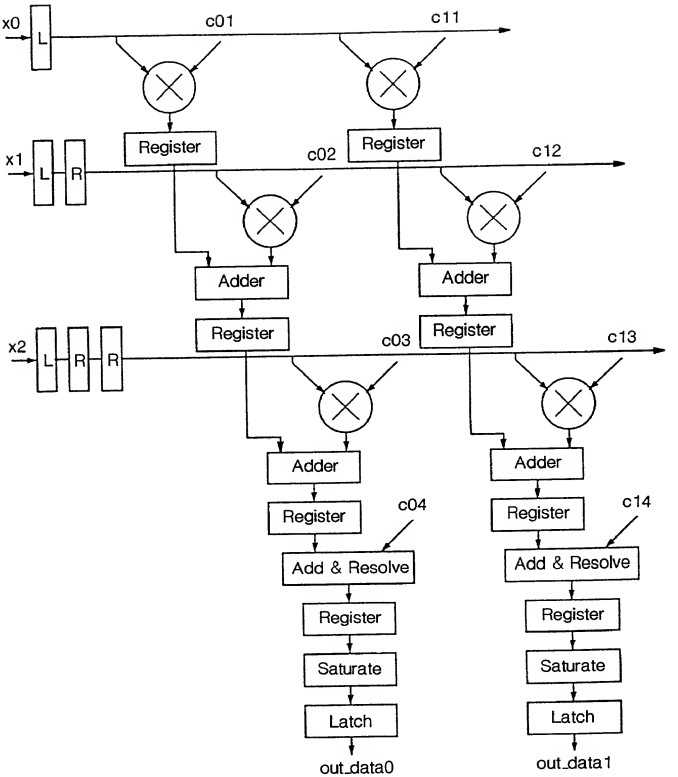

x0
L
c01
c11
Register
Register
x1
L
R
c02
c12
Adder
Adder
Register
Register
x2
L
R
R
c03
c13
Adder
Adder
Register
Register
c04
c14
Add & Resolve
Add & Resolve
Register
Register
Saturate
Saturate
Latch
Latch
out_data0
out_data1

FIG. 169